The MARS Code System User's Guide
Version 15(2004) — DRAFT

Nikolai V. Mokhov and Catherine C. James

*F*ermi National Accelerator Laboratory

P.O. Box 500, Batavia, Illinois 60510

September 13, 2004

This is a draft manual on the newest MARS15 code as of July 31, 2004. Section 4 on the input files is up-to-date. All other sections need further update. This manual along with recent principal papers describing the new features of the MARS15 code can be downloaded from the updated MARS web site *http://www-ap.fnal.gov/MARS/*:

- N. V. Mokhov, K. K. Gudima, C. C. James et al, "Recent Enhancements to the MARS15 Code", Fermilab-Conf-04/053 (2004) [1].

- N. V. Mokhov, K. K. Gudima, S. G. Mashnik et al, "Towards a Heavy-Ion Transport Capability in the MARS15 Code", Fermilab-Conf-04/052 (2004) [2].

- M. A. Kostin and N. V. Mokhov, "Parallelizing the MARS15 Code with MPI for Shielding Applications", Fermilab-Conf-04/054 (2004) [3].

- M. A. Kostin, O. E. Krivosheev, N. V. Mokhov, I. S. Tropin, "An Improved MAD-MARS Beam Line Builder: Users Guide", Fermilab-FN-738 (2003) [4].

- S. I. Striganov, "On the theory and simulation of multiple Coulomb scattering of heavy charged particles", Fermilab-Conf-04/056 (2004) [5].

It is also very helpful for users to look at recent MARS applications [6, 7, 8]

The main contributors to this version of the code are N.V. Mokhov, K.K. Gudima, C.C. James, M.A. Kostin, O.E. Krivosheev, S.G. Mashnik, I.L. Rakhno, A.J. Sierk, and S.I. Striganov.

Abstract

This paper is a user's guide to the current version of the MARS15 Monte Carlo code. MARS allows inclusive and—in many cases—exclusive simulations of three-dimensional hadronic and electromagnetic cascades, heavy ion, muon and low energy neutron-photon transport in accelerator, detector, spacecraft and shielding components in the energy range from a fraction of an electronvolt up to about 100 TeV. The code has undergone substantial improvements since the last documented version MARS13(95) and previous releases of MARS14 and all these as well as other specific features of the MARS code system are explained in detail. Descriptions of general input and output with commentary and recomendations are given. Examples are given for running the program with distributed sources, complex compounds, arbitrary geometries, and magnetic fields. A built-in powerful graphical-user interface allowing visualization and detailed analysis of the geometry description and results of simulation is decribed. Enhanced options are described for the MARS code being coupled with: the DPMJET hadron event generator, the MCNP code for low-energy neutron transport, the ANSYS system for thermal and stress analyses, physics analysis and graphics packages, the MAD accelerator lattice description via a iniversal MAD-MARS beam line builder, and the STRUCT program for tracking particles in accelerator lattices with beam loss recording. Use of the code in a multistage mode, coupled with event generators (DPMJET), with the STRUCT program for tracking particles in accelerator lattices with beam loss recording, and with physics analysis and graphics packages is demonstrated with typical input and output examples.

2

Contents

List of Tables

List of Figures

1 Introduction *Needs further editing*

The MARS code system is a set of Monte Carlo programs for simulation of three-dimensional hadronic and electromagnetic cascades, and the transport of particles including heavy ions through matter, for particles with energies ranging from a fraction of an electron volt to 100 TeV. In an inclusive mode, the program is capable of fast cascade simulations even at very high energies. The simulated particles are tracked through a user described geometry, with all the appropriate physics interactions applied. The users geometry model can be very simple to very complex, incorporating compound materials of any shape and arbitrary magnetic and electric fields. MARS tabulates the particles which pass through various portions of the geometry, and produces results on particle fluxes, spectra, energy deposition, material activation, DPA, and many other quantities. The results are presented in tables, histograms, and other specialized formats. MARS allows the user extensive control over the simulation's physics processes, tabulation criteria, and runtime optimizations, via a wide variety of options and switches implemented from an input file, and via the provided customizable user subroutines.

The basic model for the original MARS program [9, 10], introduced in 1974, came from Feynman's ideas concerning an inclusive approach to multiparticle reactions [11] and *weighting* techniques. At each interaction vertex, a particle cascade tree is constructed using only a fixed number of representative particles (the precise number and type depending on the specifics of the interaction), and each particle carries a statistical weight which is equal, in the simplest case, to the partial mean multiplicity of the particular event. Energy and momentum are conserved *on the average* over a number of collisions. A similar scheme is the basis of the program CASIM [12, 13], also introduced in 1975.

The practical reasons for the inclusive scheme as described in [14, 15, 16, 17] are:

- the CPU time per incident particle grows only logarithmically with incident energy, compared to the linear rise in the exclusive mode, and this allows for easier simulation of multi-TeV particle cascades;

- in many applications one considers effects due to the simultaneous interactions of a huge number of particles, so to describe the cascades it is sufficient to obtain the first moment of the distribution function using the inclusive cross-sections, in the same manner as with Boltzman's equation;

- experimental data on inclusive interaction spectra are more readily available than for exclusive ones;

- the use of statistical weights allows the production of a given particle type to be enhanced within the phase-space region of interest, especially for rarely produced particles.

Other codes have also adopted a similar approach to simulate very high energy electromagnetic [18] and hadronic [19] showers. A disadvantage of this approach is the impossibility of directly studying fluctuations from cascade to cascade, or of studying production correlations. So, MARS offers an alternative that allows for exclusive, or analgous, simulations of hadronic and electromagnetic cascades, but at a cost of greatly increased CPU-time per generated event.

From the basic starting point of using an inclusive approach, the MARS code has been developed over the past 26 years, and is now widely used in the high-energy physics, accelerator, and radiation shielding communities. The mathematical foundation and the physical model of the MARS system, and benchmarks from numerous past applications, are described in detail in [15, 17, 20, 21, 22]. The code is under continuous development to provide greater reliability, applicability, and usability, and the

physics models are frequently updated and bench-marked against available data. Besides the original code version [10], the milestones in code development are: MARS3 [23], MARS4 [24], MARS6 [25], MARS8 [26], MARS9 [27], MARS10 [28, 29], MARS12 [22, 30], MARS93 [31], the hydrodynamical MARS/MESA/SPHINX package [32] and a version for parallel processing, MARS12 [33].

The most recent manual was written for code version MARS13(95) [34], and updated since then in electronic form available on the MARS web sites. The purpose of this document is to be a user's guide for a new MARS, Version 15(2004), and as such reproduces the necessary sections of the previous manual, and adds detailed descriptions of all the main improvements and options. Section 2 gives an overview of the physics models implemented in MARS. Section 3 outlines the basics of how a geometry is described, what is produced as output, issues of statistics, interfaces to other codes, and how to obtain and compile the source. Further details are contained in subsequent Sections. Section 4 describes the syntax of the MARS input files, and how the items in them are used to control physics processes and describe simpler geometries. The User Subroutines, utilized to describe more complex geometries, are covered in Section 5. Sections 4 and 5 include examples. Section 11 explains how to use the Graphical User Interface, introduced in interim code version MARS13(99) but not included in the previous manual. Section 8 describes an interface between the MAD lattice description language and MARS input syntax. Section 10 describes the many output files produced by MARS, with details of the tables and values in the default output files, the contents of the histogram output, and the contents of various special output files; **continue this summary when the output and example sections are more complete**.

Substantially extended are histograming capabilities and material description, and improved computational performance. In addition to direct energy deposition calculations, a new set of fluence-to-dose conversion factors for all particles including neutrino are built into the code. MARS uses MCNP4C code for neutron transport below 14 MeV with corresponding photon, recoil proton and heavier particle production. The MARS system also includes links to the ANSYS code for thermal and stress analyses and to the STRUCT code [35] for multi-turn particle tracking in large synchrotrons and collider rings.

The MARS code has undergone substantial improvements since the last version MARS13(95) [34] for which a manual exists, and since previous releases of the MARS14 version. The main updates in the physics models are described in [36, 37, 38, 39, 40, 41, 42], and summarized in Section 2 below. For most recent publications on the current MARS15 version see *http://www-ap.fnal.gov/MARS/*. An overview of how one describes a geometry in MARS is given in Section 3, along with an overview of how MARS tracks through a geometry, and then tabulates and presents the simulation results. A discussion of variance reduction techniques is given in Section 3.8. An overview of the code, how to obtain it and the basics of using it, is given in Section 3.10.

The code is available for the Unix and Linux operating systems, and is distributed by the developers from Fermilab. The official MARS Web site is *http://www-ap.fnal.gov/MARS/*, and links there point to many recent applications of the code.

2 Physics Models

MARS simulates a variety of processes, taking into account all interactions of hadrons, leptons, photons and heavy ions during their passage through matter for energies from 100 TeV down to 100 keV for muons, charged hadrons and heavy ions, 1 keV for electrons and photons, and 0.00215 eV for neutrons. The new implementations include a new nuclear cross section library, a model for soft pion production, a cascade-exciton model, a quark-gluon string model, a dual parton model, deuteron-nucleus and neutrino-nucleus interaction models, a detailed description of negative hadron and muon absorption, and a unified treatment of muon, charged hadron and heavy ion electromagnetic interactions with matter. These improved algorithms have been benchmarked against available experimental data. This Section summarizes the techniques used to simulate the physics interactions which occur when particles pass through matter, with references to more detailed descriptions. The list of particle types employed by MARS is given in Table 3.2, located in Section 3.2. The matter content of the simulation is specified by a selection of the materials listed in Tables 3.4 and 3.4, or by user-customized compounds; further details are given in Section 3.4 and Section 5.2.

2.1 Nuclear Cross Sections

Hadron-nucleon cross sections. New compilations and parameterizations of elastic and inelastic σ_{hN} are implemented covering a hadron kinetic energy range 1 MeV$<$E$<$100 TeV. Total cross sections, σ_{tot}, from 1 MeV to 10 GeV for p, n, π^+ and π^- are as predicted by the new improved algorithm [43] of the Cascade-Exciton Model (CEM) [44] code CEM95 [45] while for K^+, K^- and $\overline{p}$ data compilations are used [?]. Parameterizations from [?] are relied upon for all particles between 10 GeV and 100 TeV. Elastic cross sections, σ_{el}, from 10 MeV to 10 GeV for p, n, π^+ and π^- are likewise from [43] with interpolation of data [?] for K^+, K^- and $\overline{p}$. Parameterizations from [?] are used between 10 and 200 GeV. For energies 200 GeV$<$E$<$100 TeV, the optical theorem with 'universal slope' [46] is applied. Fig.1 shows comparison of data and MARS results on σ_{tot} and σ_{el} for $\pi^- p$ collisions.

Add on heavy-ion cross sections here!!!

Hadron-nucleus cross sections. New compilations, parameterizations and integration algorithms for total, inelastic, production and elastic σ_{hA} are introduced into the code. Total, inelastic and elastic cross sections from 1 MeV to 5 GeV are described using new compilations and improved interpolation algorithms [47, 48]. At higher energies (5 GeV$<$E$<$100 TeV), σ_{tot}, σ_{in}, σ_{prod} and σ_{el} are calculated in the framework of the Glauber multiple scattering theory with the above σ_{hN} as an input. The nucleon density distribution in nuclei is represented as the symmetrized Fermi function with the parameters of [49] for medium and heavy nuclei ($Z > 10$) and the ones of [50] for $Z < 10$. An example is shown in Fig. 2 for neutron-nucleus σ_{tot} as calculated with this algorithm (solid line) and with the improved algorithm [48] (dashed line).

Photon-nucleus cross sections. Data compilation and interpolation algorithm for $\sigma_{\gamma N}$ with phenomenological A-dependence for $\sigma_{\gamma A}$ are as described in [51].

2.2 Hadron Production

This section needs more detail, enough that the casual reader gets a better idea of what the code does - a short version of sections 2 and 3 from Conf-04-053, appropriately updated, would work.

Cascade-exciton model code CEM2003. A version of the Cascade-Exciton Model of nuclear
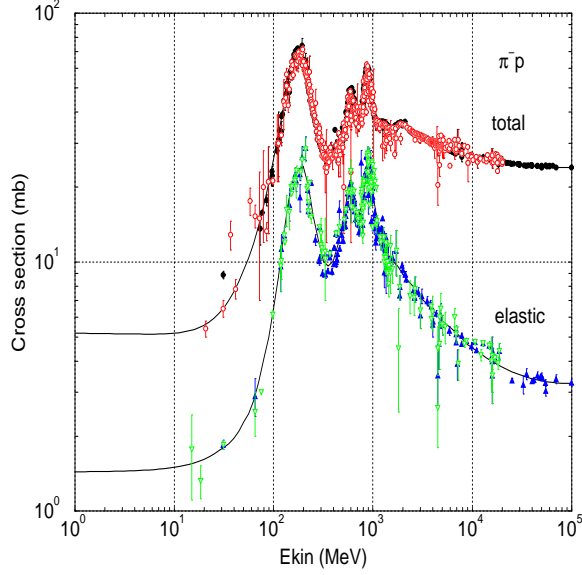
Figure 1: MARS cross sections in comparison with experimental data: σ_{tot} and σ_{el} for $\pi^- p$ collisions as a function of pion kinetic energy
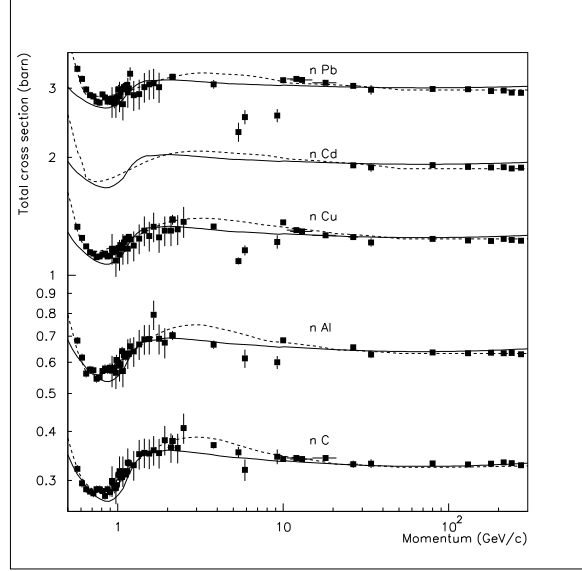
Figure 2: MARS cross sections in comparison with experimental data: σ_{tot} for neutrons *vs* beam momentum.

reactions [44] as realized in the code CEM2003 [45] and containing also several recent refinements [43] is implemented as default for 1 MeV < E < 5 GeV. The 1994 *International Code Comparison for Intermediate Energy Nuclear Data* has shown that CEM95 adequately describes nuclear reactions at intermediate energies and has one of the best predictive powers for double differential cross sections of secondary particles as compared to other available models. Besides that, it adds to MARS reliable π^--capture description (with a few modifications, e.g., radiative capture $\pi^- p \rightarrow n\gamma$), better description of photon induced reactions in the intermediate energy range and of radionuclide production. To be usable in MARS, the CEM code is converted into double precision along with some other necessary modifications. Several examples of the CEM predictions compared with experimental data and results of several other models are given in Figs. 3 and 4. One can see that on the whole, the code reproduces quite well not only spectra of secondary nucleons but also excitation functions for the spallation yields, a much more *difficult* characteristic of nuclear reactions to be predicted by any theory, and is consistent with other well-known models [52].

Quark-Gluon String Model code LAQGSM2003. Comes here !!!

Inclusive hadron production from 5 GeV to 100 TeV. An inclusive approach is the default in this energy range. The hadron production model uses a combination of phenomenological models, parameterizations and integration algorithms, covering a hadron kinetic energy range of 1 MeV<E<100 TeV as described in Refs. [37, 53, 54]. An improved phenomenological model has recently been developed and introduced into MARS as the default to describe pion production in high-energy proton-nucleus interactions [55]. Special attention is paid to high-p_t events, π^0 and kaon production, and low-momentum pions (p<2 GeV/c) from intermediate incident proton momenta (5< p_0 <30 GeV/c). The following form is used for the double differential cross section of the $pA \rightarrow \pi X$ reaction:

$$\frac{d^2\sigma^{pA\rightarrow\pi X}}{dpd\Omega} = R^{pA\rightarrow\pi X}(A, p_0, p, p_\perp)\frac{d^2\sigma^{pp\rightarrow\pi X}}{dpd\Omega} \tag{1}$$

where p and $p_\perp$ are total and transverse momenta of π, and A is the atomic mass of the target nucleus.
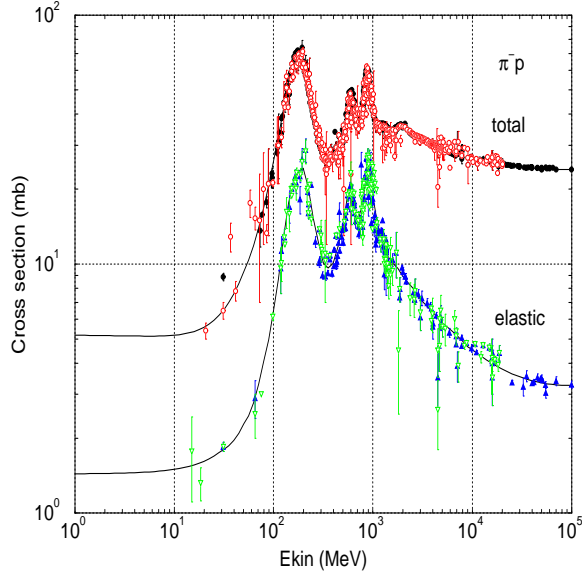
Figure 3: missing figure from CEM section
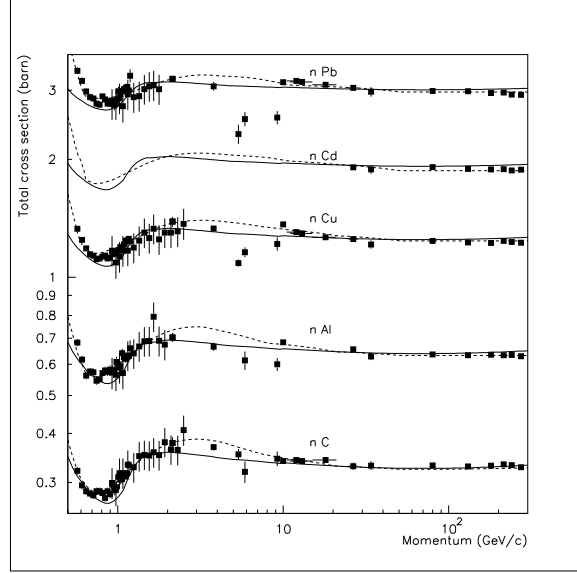
Figure 4: missing figure from CEM section

The function $R^{pA \to \pi X}$, measurable with much higher precision than the absolute yields, is almost independent of $p_\perp$ and its dependence on p_0 and p is much weaker than for the differential cross-section itself. Rather sophisticated algorithms have been developed to treat this function for pion production on nuclei in the forward ($x_F > 0$) and backward ($x_F < 0$) hemispheres separately. It is demonstrated in [55] that model predictions are in a good agreement with data in the entire kinematic region. Typical examples of comparison with data are shown in Figs. 5 and 6. Calculations with the MARS13(98) code of the pion double differential spectra from a thick lead target at p_0=8 GeV/c agree reasonably well with data [56] in the *difficult* momentum region 0.5< p <5 GeV/c, Fig. 7 whereas GEANT seems to have a problem. Information on the nuclides generated in nuclear collisions is now scored, or accumulated, and reported in the output results.

Deuteron-nucleus collisions. Deuteron interactions have little in common with the general picture of the interaction between complex nuclei because of the deuteron's relatively large size and small binding energy. Therefore a special model has been developed [57] and implemented into MARS. Deuteron-nucleus interactions are classified as elastic, dissociation, stripping, and full inelastic. In elastic interactions the deuteron emerges intact in the final state while the nucleus may be unchanged (coherent elastic) or have lost one nucleon (incoherent). Coherent elastic uses Glauber's treatment with some adjustments of the parameters to fit experiment. Incoherent elastic scattering assumes a differential cross section to be twice that of the proton—using the prescription of [58]—and the nuclear parameters as for the coherent case. This is then multiplied by a deuteron– and a nucleon form factor as well as a Pauli suppression factor which hinders low momentum transfers. Exchange of a long range virtual photon may result in Coulomb dissociation whereby the deuteron splits into a proton and neutron. This is calculated using a Weiszacker-Williams approach for virtual photon emission. Dissociation may also result from (nuclear) elastic processes at relatively high momentum transfers. In stripping one nucleon undergoes an inelastic nuclear event while its partner continues without interaction. The total stripping probability is calculated based on the projected n–p separation as predicted by the deuteron wave function [59] and geometrical arguments. Deuterons dissociate as in [60] with full relativistic kinematics. Interaction with the nucleus by one of the partners proceeds within the standard MARS scheme. In full inelastic events both nucleons interact with the nucleus. The stripping routine

13
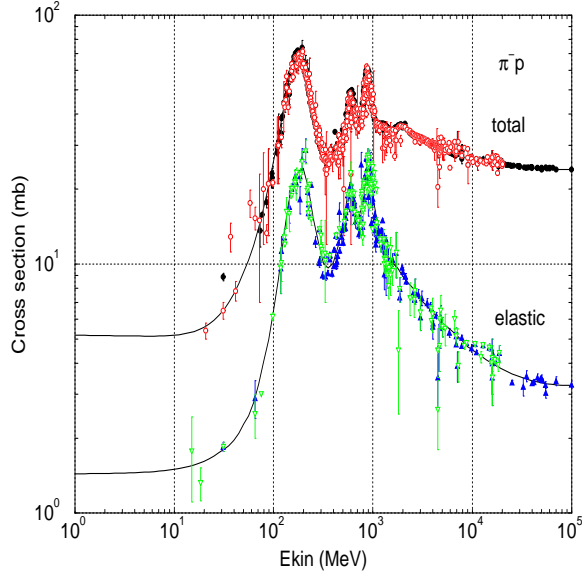
Figure 5: missing figure from hadron production section
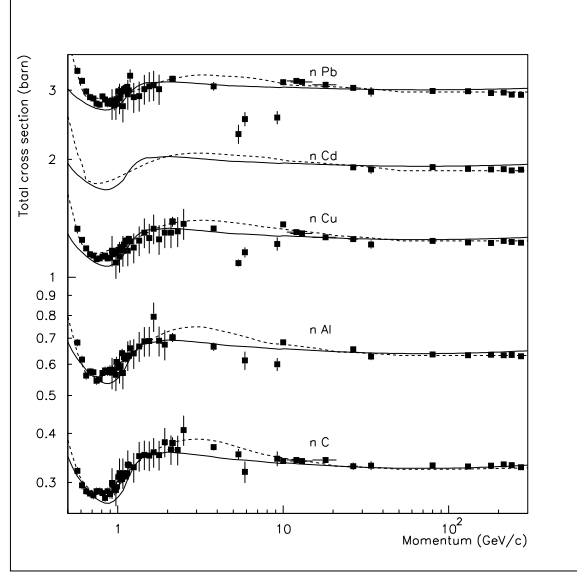
Figure 6: missing figure from hadron production section

provides the angular deflection and momentum of each nucleon after which both are allowed to interact as other MARS nucleons. As an example, a calculated π, K-meson yield out of a 3-cm radius gallium target 36-cm long in a 7.5-cm radius solenoid (B=20 T) is presented in Fig. 8 for proton and deuteron beams of equal momentum per nucleon.

Quark-Gluon String Model code LAQGSM2003. Exclusive particle event generator for hadron and heavy-ion nuclear interactions from 1 MeV to 1000 GeV. Comes here !!!

Exclusive hadron production from 5 GeV to 100 TeV. The DPMJET-3.0 code [61, 62] was recently implemented into MARS to sample the initial hN, hA, AA and νA interaction for 5 GeV<E<100 TeV [37]. This provides—at least partially—features of a full exclusive event generation with all known particles in a final state. The DPMJET code has been proven to be consistent with collider and cosmic ray data in a multi-TeV energy region. The two-component Dual Parton Model is used with multiple soft chains and multiple minijets at each elementary interaction. Within this model the high energy projectile undergoes a multiple scattering process as formulated in the Glauber approach. Particle production is realized by the fragmentation of colorless parton-parton chains constructed from the quark content of the interacting hadrons. The code includes cascading of secondaries—suppressed by the formation time concept—within both target and projectile nucleus. The excitation energies of the remaining target– and projectile nuclei are calculated and simulation of subsequent nuclear evaporation is included in the model. The coupling of these new features to the MARS code is very CPU-time consuming and is used optionally only.

2.3 Elastic Scattering

Modern evaluated nuclear data as well as fitting formulae are used to simulate hadron-nucleus elastic scattering. For protons nuclear, Coulomb elastic scattering and their interference is taken into account. At E>5 GeV, a simple analytical description used in the code for both coherent and incoherent components of $d\sigma/dt$ is quite consistent with experiment.

14
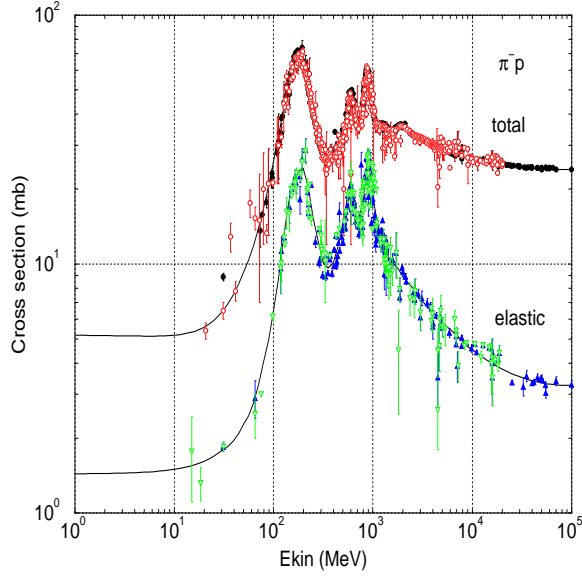
Figure 7: missing figure from hadron production section
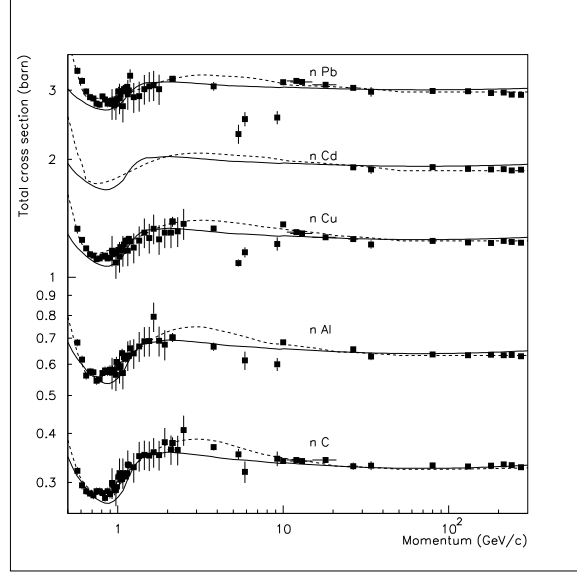
Figure 8: missing figure from deuteron-nucleus collision section

2.4 Muon Production

Simulation algorithms for $\pi \to \mu\nu(\tilde{\nu})$ and $K \to \mu\nu(\tilde{\nu})$ decays and for prompt muon production (single muons in charmed meson decays, $\mu^+\mu^-$ pairs in vector muon decays, and the dimuon continuum) with forced generation of weighted muons have been improved. Prompt muons produced in electromagnetic showers are described in detail.

- Bethe-Heitler pairs $\gamma Z \to Z \mu^+ \mu^-$ are produced at $E_\gamma \geq 0.25\,\text{GeV}$ at a rate of $(m_e/m_\mu)^2$ times that for e^+e^- with the appropriate statistical weights and a complete simulation of electromagnetic showers. It was shown [63] that this approach produces remarkable results that agree with those based on the numerical integration in the Tsai formalism [64].

- muon production with forced decays of mesons and short-lived resonances [15, 17, 65, 66];

- very efficient algorithms for muon interactions (ionization, bremsstrahlung, direct e^+e^- pair, and deep inelastic) and transport [67] well advanced compared to previous versions [28, 17, 65, 66];

- optional forced $\mu \to e\nu\tilde{\nu}$ decays and synchrotron radiation generation [68];

- At $E \geq 45\,\text{GeV}$ direct positron annihilation $e^+e^- \to \mu^+\mu^-$ is simulated according to [69] with $\sigma = 86.8/s$ nb, where s is in GeV^2, and with $(1 + cos^2\theta)$ as the angular distribution.

- For $\mu \to e\nu\tilde{\nu}$ decays, the vector momenta of the emitted electrons and neutrinos are sampled according to the differential decay probability of the *Vector–Axial* model of four-fermion interactions,[69].

2.5 Electromagnetic Interactions of Heavy Particles

Electromagnetic interactions of muons, charged hadrons and heavy ions in arbitrary composite materials are simulated down to several tens of keV. Radiative processes and atomic excitation and

ionization with energy transfer ϵ greater than a cutoff ϵ_c are considered as discrete events involving production of δ-electrons, e^+e^--pairs, and bremsstrahlung photons which are followed explicitly [70]. Energy losses with $\epsilon < \epsilon_c$ (so-called restricted losses) are considered as continuous. Their distribution is described by Vavilov function—with redefined parameters—which approaches a Gaussian with decreasing ϵ_c. Independent of energy, material or thickness traversed, the quality of the Gaussian approximation is governed by the average number of events (κ_n) one chooses to evaluate individually and becomes acceptable for most purposes when $\kappa_n > 10$. Bremsstrahlung and direct e^+e^- production differential cross-sections used in the code are as given in Ref. [71].

Multiple Coulomb scattering is modeled from the Moliere distribution with nuclear form-factors included [72]. A very careful treatment is done in MARS of processes near and below the Coulomb barrier in hadron, muon and heavy ion transport (ionization absorption vs nuclear interaction vs decay) as is further described in Ref. [37]. The scattering is applied as a continuous process while the particle passes through material, rather than being applied at discrete locations.

2.6 Electromagnetic Showers

New modules for simulating electromagnetic showers based on current knowledge of physics of electromagnetic interactions were recently developed and have been implemented into the code [73]. The main focus is given to electron and photon interactions in arbitrary composite solid, liquid and gaseous materials at low energies (1 keV to a few MeV). The entire shower, and such processes as emission of synchrotron photons, photohadron production, $\gamma Z \to \mu^+\mu^-$ and $e^+e^- \to \mu^+\mu^-$, can be treated—in the spirit of the MARS framework—either analogously or inclusively with corresponding statistical weights. The choice of method is left for the user to decide, via the input settings.

following statement needs more detail Generation and transport of de-excitation photons is improved. Undesirable fluctuations in the inclusive description of electromagnetic showers are reduced.

2.7 Synchrotron Radiation *to be added*

2.8 Stopped Hadrons and Muons

A very careful treatment is done in MARS of processes near and below the Coulomb barrier in hadron and muon transport (ionization absorption vs nuclear interaction vs decay).

Pions. A stopping π^+ decays into μ^+ of 4.1 MeV plus a neutrino while a π^- attaches to a nucleus (via the modified Fermi-Teller law). While cascading down the atomic energy levels, the pion is captured from a high orbit thus emitting only a few low energy photons which are neglected here. The hadronic interaction of the stopped π^- is treated using the Cascade-Exciton Model [45] with a few modifications. When hydrogen is the target it is assumed there is a 60% probability to for charge exchange ($\pi^- p \to \pi^0 n$) whereupon the π^0 decays into two photons of 68.9 MeV each and the neutron acquires a small (0.4 MeV) kinetic energy. The remaining 40% of stopped π^- in hydrogen interact via radiative capture: $\pi^- p \to n\gamma$. Here the photon acquires 129.4 MeV and the neutron 8.9 MeV kinetic energy. Other nuclides have a much smaller probability for radiative capture (1–2% which is taken into account in competition with CEM95). The photon energy is chosen from an empirical fit to experiment while the remainder is deposited as excitation energy.

Muons. A stopping μ^+ always decays into $e\nu\overline{\nu}$ while a μ^- attaches itself to a nucleus. When a μ^- stops in a compound or mixture one first decides to which nucleus the μ^- attaches (modified

Fermi-Teller law). Following attachment the muon may still decay as decided by comparing capture and decay lifetimes of which the latter is favored for light nuclei (Z≤11). A captured μ^- then cascades down to the ground state of the muonic atom emitting photons along with some Auger electrons, all of which is simulated using approximate fits to the atomic energy levels. In hydrogen muon capture always produces a 5.1 MeV neutron via inverse β-decay. In complex nuclei the giant dipole resonance plays a role and results in an 'evaporation'-type neutron spectrum with one or more resonances superimposed. This is illustrated in Fig. 9 which shows the neutron spectrum resulting from μ^- capture on oxygen. In addition smaller numbers of evaporation-type charged particles and photons may be emitted. Calculated with the above algorithms longitudinal dose distributions in a slab tissue-equivalent phantom are shown in Fig. 10 at the axis of 150 MeV proton and 75 MeV pion, muon and neutron beams striking the phantom.
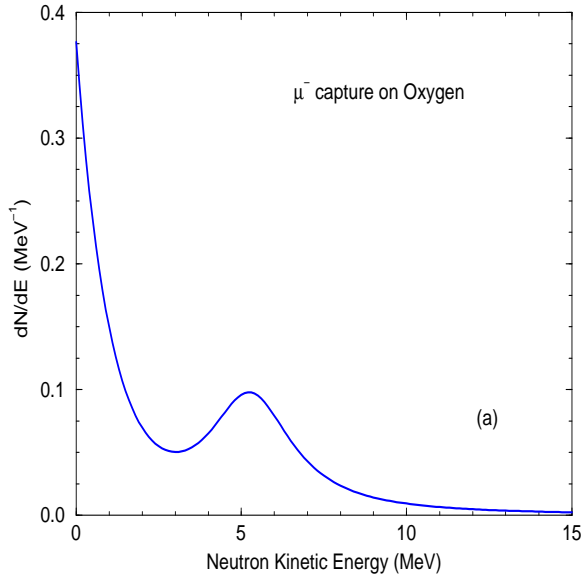
Figure 9: Neutron spectrum generated in a μ^- capture on oxygen atom
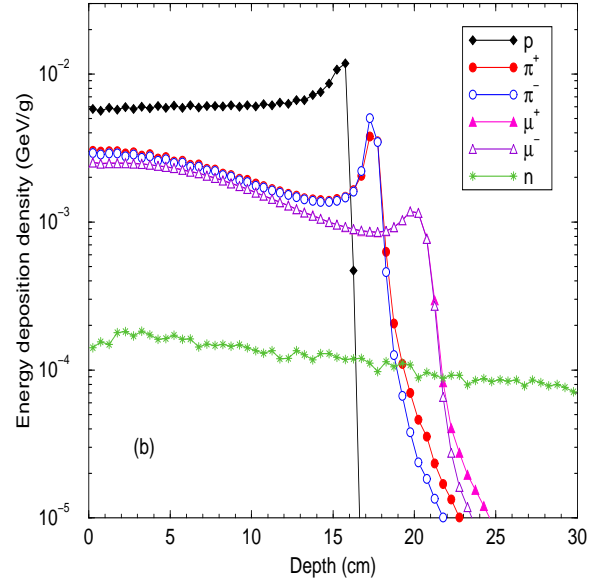
Figure 10: Axial absorbed dose in a tissue-equivalent phantom for a 150 MeV (p) and 75 MeV (π, μ, n) 1×1 cm beams.

Antiprotons. Stopped $\overline{p}$ attach to nuclei in the same way as π^- or μ^-.Annihilation at rest is assumed to produce only pions, neglecting some of the rarer modes involving strange particles. Charges of produced pions are slightly skewed towards π^- in view of the 'brought in' negative charge. Pion momenta are chosen from an inclusive distribution loosely based on experiment. The energy weighted distribution is normalized to twice the nucleon mass which predicts a multiplicity of 4.3 —close to observation. In a complex nucleus the annihilation is treated as though it occurs on free nucleon except that each pion produced by the annihilation process is given a 50% probability to interact within the nucleus. This shortcut attempts to include —at least qualitatively —participation by the constituents nucleons.

For antiprotons in flight the annihilation cross section results in a larger cross section for $\overline{p}A$ vis-a-vis pA, especially for light nuclei at lower energies.Total cross sections for both $\overline{p}A$ and pA are estimated on the basis of simple geometrical considerations and $\overline{p}p$, $\overline{p}n$ and pp,pn cross sections.The ratio $\sigma_{\overline{p}A}/\sigma_{pA}$ is then applied to the more accurate σ_{pA} used in MARS. Annihilation in flight uses the same inclusive pion distribution as at rest in the $\overline{p}$-nucleon rest frame after which the pions are Lorentz transformed back to the lab. Above about 0.1 GeV/c a small $\overline{p}p \to \overline{n}n$ component is included.

For both mechanisms *nuclear* target effects are again approximated by allowing emerging particles to interact in the same nucleus or escape each with one half probability. There is also added a third component in which the $\overline{p}$ or $\overline{n}$ interact only quasi-elastically with the nucleons. These are simulated using conventional MARS algorithms exactly as for protons except that the fastest nucleon emerging (leading particle) from the collision is identified as its antiparticle.

2.9 Neutrino Interactions

A special weighted neutrino interaction generator has been developed [37, 74] and incorporated into MARS. This model permits the selection of the energy and angle of each particle (ν, e, μ and hadrons) emanating from a simulated interaction. These particles, and the showers initiated by them, are then further processed in the code in the usual way. Four types of neutrinos are distinguished throughout ($\nu_\mu, \overline{\nu}_\mu, \nu_e, \overline{\nu}_e$) and the model identifies all possible types of neutrino interactions with nuclei. The corresponding formulas for these processes as well as results of Monte Carlo simulations for muon colliders and storage rings are described in [74].

2.10 Low Energy Neutrons

The default low energy neutron model in MARS is sufficient for most applications. It uses a 28-group cross-section library, which in turn is derived from data on a set of 14 materials. The cross-sections are extrapolated to cover other materials not in the initial set of 14. Once the energy of neutrons falls below 14 MeV, subsequent neutron interactions are described using MCNP4C [75] subroutine modules, which are part of the default model. These modules model recoil protons, and heavier recoils and photons, which result from the thermal neutron capture on ^6Li and ^{10}B, and allow a detailed description of corresponding effects in hydrogenous, borated and lithium-loaded materials [76]. However, $n-/gamma$ reactions which occur when the neutron has stopped are not modeled.

In some cases more precise modeling of low energy neutrons is necessary, in particular when predicting residual dose rates, and when predicting prompt dose rates in labyrinth type geometries. Residual rates are sensitive to the materials present, and prompt dose rates to the $n-/gamma$ reactions which occur when slow neutrons are stopped and captured. In these cases, MARS can be used coupled with the full MCNP4C code: MARS generates the initial neutron but sends it to MCNP4 for tracking and interactions, once tracked to capture or to it's cutoff energy, the particle information is handed back to MARS for compilation of tallies and results. MCNP code contains very detailed material handling, taking into account the nuclear physics details that occur in interactions with different nuclei, and this affects residual rate calculations. MCNP also includes n-gamma reactions when the neutron is finally stopped and captured. These reactions affect the prompt dose calculations in labyrinth type geometries where slow neutrons are dominant.

Using MARS in this second mode, coupled with the full MCNP code, requires that the user obtain and install the MCNP code library. For more information see Section 6.

2.11 Analogous-Inclusive Control

All electromagnetic and weak process and many of strong interactions can be treated either analogously or inclusively with corresponding statistical weights. The choice of method is left for the user to decide, via the input settings.

3   Using the MARS Package *updated 7/2004*

MARS uses it's physics models to generate particles and track them through a described geometry, applying appropriate physics interactions, and tabulating the particle flux, energy deposition, and other parameters in the various portions of the geometry. This Section describes how the MARS package functions, by giving overviews of : MARS geometric zones; the particle types used in the simulation and how MARS tracks these particles through a modeled geometry; how materials are defined; how results are tabulated and what histogramming options are available; variance reduction techniques; and interfaces to other codes.

3.1   Overview of Geometric Zones

The geometry description, or model, is based on a contiguous array of volumes called zones. As particles are tracked through the model, MARS queries specific subroutines to determine which zone the particle is in. Each zone is uniquely numbered, and the zone number corresponds to the array index used by many arrays in the program. In particular, there is an array, indexed by zone number, which tell MARS what material is present in each zone, and the material in turn is used to define many parameters which affect the physics interactions which can occur within each zone. As particles are generated and tracked, the *path length* of each particle type within each zone is accumulated and stored; the final *path length* divided by the zone volume results in the flux for each particle type through each zone. Particle flux, tabulated by zone number, is used in turn to calculate many other results from the program. In short, by controlling the size, location and material specifications for the zones, the user controls the level of detail in the results which MARS reports. The user can also tweak parameters, from zone to zone, which control some of the interactions which MARS simulates.

MARS classifies zones as one of three types: Standard, Extended and Non-Standard. Standard zones are those whose borders are defined by quantities entered in the MARS.INP input deck. Standard zones must have a cylindrical symmetry, with borders defined only on the z, r and ϕ axes; the syntax is described in Section 4.2.4. Extended zones are those whose borders are defined by the contents of the GEOM.INP input deck; the syntax for this deck is described in Section 4.4. Extended zones are similar to the geometry description used by the Geant Monte Carlo, in that they are constructed from a set of contiguous or overlapping geometrical shapes such as boxes, spheres and cones. Non-Standard zones are those described by the user's FORTRAN encoded description of the geometry. There is no restriction on the shape of Non-Standard zones; they can have cylindrical or cartesian symmetry, or can be of an arbitrary shape too intricate to render using the Extended syntax. Further details on setting up non-Standard zones is in Section 5.4.

Any type of zone can hold any material which the user has declared to be "in use" (see Sections 3.4 and 4.2.3). But there can be only one material in each zone. On the other hand, a magnetic field can vary from point to point within a zone. The difference between the requirements for materials and fields within zones has to do with how the tracking is handled, described below in Section 3.3.

The entire volume of the user's model must be filled by either Standard, Extended or Non-Standard zones; there can be no "holes" left undefined and yet surrounded by defined spaces. If such holes exist, then any simulated particle which enters the volume of the hole will be dropped from the tracking list, not propagated into the surrounding zones, and the simulation results will not be correct. There is, however, a specific material called "black hole" which MARS uses to fill in undefined spaces around the outer edges of a model, and it can show up as black fill in the GUI visualization of the model. There are also other uses for the "black hole" material discussed elsewhere.

19

The outermost extent of the user's entire model is defined by MARS to be the outermost extent of the largest Standard zones, in Z and R, and this is given by the maximum geometrical values in the ZSEC and RSEC cards in the MARS.INP input deck. This fact implies that there must be at least one Standard zone - a sort of container zone. All other zones, Standard, Extended and non-Standard, must fit within it or have outer boundaries aligned with it. For example, if the users model is defined entirely using FORTRAN encoded non-Standard zones, he must still have at least one Standard zone, and this Standard zone must be sized to surround the maximum geometrical extent of the encoded non-Standard zones.

The total number of all zones, Standard, Extended and non-Standard, has a maximum value of 100,001, and this limitation is set by the maximum size of the arrays used to store and accumulate the results of the simulation. The number of Extended zones must be between 0 and $50,000$. There is no set limit on the number of Standard and non-Standard zones, except the upper limit on the sum of all zone types. It is up to the user to keep track of the numbers of the zone types declared, and to be certain they remain within the stated limits.

An interactive GUI interface is included within MARS, which allows the user to view his geometry, and check items such as zone boundaries, the material content of zones, and the zone index number assignments. Details on using the GUI interface are given in Section 11.

3.2   Particles used in MARS

The list of particles which the MARS simulation produces, interacts and tracks through a model is given in Table 3.2. Heavy ions are now transported in the code. The mesons listed, π^+, π^-, K^+, K^-, and muons are allowed to decay in flight according to their lifetimes, if they survive interactions in matter between their production and decay points. In an inclusive default non-LAQGSM mode all other particle types generated at an interaction point are forced to decay immediately, using their most probable branching fractions, producing daughters which are listed in Table 3.2. While this model for decays is appropriate for π^0, ρ, ω, D, and J/Ψ and other short-lived states, for particles with longer lifetimes, such as K^0, Λ and other hyperons, it is not. As a result the flux of π-mesons near the production point for these particles may be slightly larger than reality. Therefore, it has been corrected for the current version MARS15. In the exclusive LAQGSM mode hyperons, antineutrons etc are treated as all other particles with accurate treatment of decays and all other interaction processes.

Each particle in Table 3.2 has an index associated with it. This index is used to refer to the particle type in various places in the MARS code. For example, the primary particle type can be specified from the input MARS.INP file using this index; and the user histogram subroutines can utilize the index to accumulate data on a particular particle type. In the MARS common blocks and the user subroutines, the index value is generally assigned to variable JJ.

Heavy ions with $A > 4$ have the following indicies: $ID = 1000 * Z1 + A1 - Z1$.

3.3   Tracking

The methods used to simulate the passage of particles through matter are a large part of any physics Monte Carlo system and MARS is no exception. The path any particle takes through matter has both discrete and continuous components. An example of a discrete process is nuclear collisions; an example of a continuous processes is the trajectory of a particle through a magnetic field. In the MARS

Table 1: Transported particle types and their corresponding index value.

1	2	3	4	5	6	7	8	9	10	11	12
p	n	π^+	π^-	K^+	K^-	μ^+	μ^-	γ	e^-	e^+	$\tilde{p}$
	13	14	15	16	17	18	19	20	21		
	π^0	d	t	He_3	He_4	ν_μ	$\tilde{\nu}_\mu$	ν_e	$\tilde{\nu}_e$		
	22	23	24	25	26	27	28	29	30	31	
	K0L	K0S	K0	AK0	LAM	ALAM	SIG+	SIG0	SIG-	ANEU	
	32	33	34	35	36	37	38	39	40		
	KSI0	KSI-	OM-	ASI+	ASI0	ASI-	AKSI0	AKSI-	AOM-		

simulation, a particle's trajectory is approximated by a series of connected line segments. The possibility of a discrete interaction occurring, or the effect of a magnetic field, is evaluated at the ends of a segment; the length of each segment and the magnitude of the change in direction from segment to segment are what control the overall accuracy of the tracking simulation.

Therefore, the first thing MARS does when tracking a particle is determine the size of the next line segment. There are four main parts to this determination. First, MARS calculates the mean distance to the next discrete process, D_{disc}, where the set of these discrete processes consists of nuclear inelastic and elastic interactions, particle decay, and energy loss due to knock-on electrons. This distance depends upon the particle type and energy, and on the material the particle is passing through. Second, MARS calculates the step size D_{cont} allowed by the algorithms which model the two main continuous processes, multiple coulomb scattering and the effect of magnetic fields. The default step size for the scattering is $0.0001/\rho$ cm, where ρ is the density of the current material in g/cm^3. The step size for tracking through magnetic fields varies with the particle energy, as it is determined by limits on the bend angle; the MARS defaults for these limits are adequate for most situations, and can be modified by the user using the ALMX card in the input deck, described in Section 4.2.7. Third, MARS sets the overall interaction distance D_{int} by choosing the minimum of D_{disc} and D_{cont}.

The fourth part of the line segment size determination compares D_{int} to the step size allowed by the geometry description, D_G. The geometry portion of the tracking steps through the users model using a "pilot" step size, and after each such step, MARS queries the geometry description routines to determine what material is present at the end point of the step. If the material has not changed from the previous step then D_G is set to the pilot step size. If however the material has changed, then the code jumps back to near the start of the pilot step, checks to see that the initial material is located there, jumps forward again, with a reduced step size, checks to see if the material has changed, and so back and forth in smaller and smaller steps until the smallest allowed step size bridges across the boundary between the two materials. Figure 11 shows a simple schematic of this boundary localization procedure. D_G is reset at each iteration to a shortened step length which approaches or crosses the material boundary, and is compared to D_{int}. The boundary localization process halts when D_G is smaller than D_{int}, or when D_G has reached an allowed minimum size. If this allowed minimum step crosses the boundary into a new material, then D_{int} must be redefined, and the tracking sequence resets back to part one described above. The final tracking line segment length is the minimum of D_{int} and D_G.

The accuracy of the geometric boundary localization process is controlled by the user, using the input deck as described in Sections 4.2.1 and 4.2.3. The pilot and minimum step sizes can be defined separately for different materials, using the *material dependant* cards. *Global* pilot and minimum step
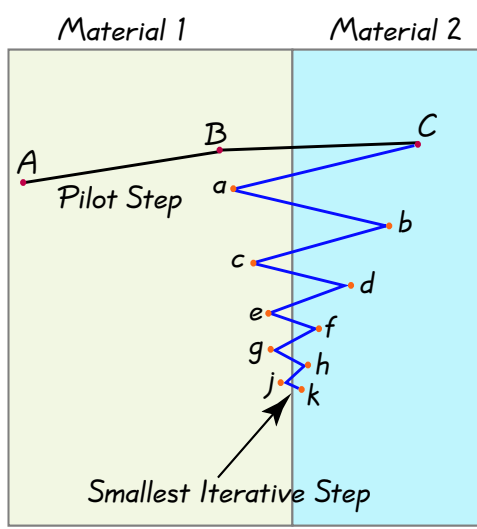
Zone Boundary Localization

The initial geometric line segment length, D_G, is the Pilot Step Length. If the material type changes after taking a pilot step, as between points B and C, then the code iterates back and forth, as in points a through k, stopping when the length between two steps is the smallest allowed iterative step size.

The iterative steps are expanded here for clarity. They actually all lie on top of each other, along the line connecting B and C.

Figure 11: A schematic of the boundary localization procedure.

sizes are otherwise applied. Guidelines for setting the values of these step sizes to attain the desired accuracy are also discussed in Sections 4.2.1 and 4.2.3, and the example in Section 4.6.3 discusses in detail the correct application of the step size controls.

Another user control over the tracking is the particle threshold energy. As particles pass through matter they lose energy. The user can specify an energy below which particles will be dropped from the tracking; the main advantage of this feature is to reduce the computing time spent tracking particles which will have little or no effect on a particular result. The specification is made via the input deck. Just as for boundary localization, there are *global* and *material dependant* thresholds. The *global* energy thresholds are applied in all materials, although there are several of them, so that each class of particle types can have a separate threshold applied, as described in Section 4.2.2. The *material dependant* thresholds are applied only where those materials are present, and there are also separate thresholds for each class of particle types, as described in Section 4.2.3.

Details on the development of MARS particle tracking methods can be found in references [17, 77, 78].

3.4 Materials

The user declares what materials are present in his simulation via the input file MARS.INP. The syntax is described in Section 4.2.3 and utilizes a 2-4 character string as an identifier. A maximum of 50 materials can be used within any simulation. An index, 1:50, is assigned to each declared material according to the order in which they are listed by the user in the MARS.INP file, and the user must use these indicies when assigning a specific material to a geometric zone. There can be no more than one material within a single geometric zone. A material can be declared multiple times, so that different step-size or energy thresholds can be applied to the same material; and example of this is given in Section 4.6.3.

There are two special materials within the MARS code. The first one, *blackhole* is not included in the total number and not declared in MARS.INP. The program assigns IM$= -1$ and region number

$N \leq 0$ to *blackholes*, which are either all material outside of the simulated volume ($N = 0$) or any region inside marked with $N = -1, -2, -3, ...$ where entering marticle is immediately killed. Usefull for special scoring or forming a source term for the cosequent runs, both in subroutine LEAK. The second one, *vacuum* can be used either the same way, i.e. not included in the total number, the program assigns IM= 0 (usually it is a global vacuum), or as any other materials listed using (multiply if needed) the name VAC with corresponding IM> 0. Last option is useful if one needs to assign different transport properties (step parameters and cutoff energies) to the different vacuum regions.

The first 100 elements in the periodic table, and many common compounds used in particle and nuclear physics, are built-in to the MARS code, with pre-defined character identifiers and properties. These are listed in Tables 3.4 and 3.4. The user can also define his own compounds, or mixtures, as described in Sections 4.2.3 and 5.2. All compounds are defined through the weight or atomic fractions of the individual elements they consist of. At program startup, MARS calculates the cross-sections and other physics interaction parameters for the list of compounds declared for use in the simulation. The precise effect of individual elements in compounds is then taken into account for all the electromagnetic and nuclear elastic and inelastic processes modeled by MARS; homogenization (averaging) of materials is no longer performed, as was allowed in previous versions of MARS (MARS-13-95 and earlier).

The calculated cross-section tables, as well as a summary of the composition and properties of all declared materials, is written to the output MARS.OUT file. If the user has any questions about the detailed properties of a built-in material, he can declare the material, and run MARS for a single event, and then examine the information in the MARS.OUT file. The default densities for most elements are those at standard room temperature and pressure.

Several elements can now be treated in MARS15 different phases. Gaseous states are
$H, HE, N, O, F, NE, CL, AR, KR, XE, RN, AIR$
as listed in Table 3.4. Liquid states are
$LH, LHE, LN, LO, LF, LNE, LCL, LAR, LKR, LXE, WATR, CH4$
as listed in Table 3.4. Graphite can also be used in two forms: compact C with default density of 2.265 g/cm^3, and graphite $GRPH$ with default density of 1.7 g/cm^3 .

Table 2: Built-in elements. The first 100 elements in the periodic table are built-in, and listed in order of atomic number in the table below. They are represented by their standard abreviations.

'H'	'HE'	'LI'	'BE'	'B'	'C'	'N'	'O'	'F'	'NE'
'NA'	'MG'	'AL'	'SI'	'P'	'S'	'CL'	'AR'	'K'	'CA'
'SC'	'TI'	'V'	'CR'	'MN'	'FE'	'CO'	'NI'	'CU'	'ZN'
'GA'	'GE'	'AS'	'SE'	'BR'	'KR'	'RB'	'SR'	'Y'	'ZR'
'NB'	'MO'	'TC'	'RU'	'RH'	'PD'	'AG'	'CD'	'IN'	'SN'
'SB'	'TE'	'I'	'XE'	'CS'	'BA'	'LA'	'CE'	'PR'	'ND'
'PM'	'SM'	'EU'	'GD'	'TB'	'DY'	'HO'	'ER'	'TM'	'YB'
'LU'	'HF'	'TA'	'W'	'RE'	'OS'	'IR'	'PT'	'AU'	'HG'
'TL'	'PB'	'BI'	'PO'	'AT'	'RN'	'FR'	'RA'	'AC'	'TH'
'PA'	'U'	'NP'	'PU'	'AM'	'CM'	'BK'	'CF'	'ES'	'FM'

The general chemical makeup of the built-in compounds is listed in Table 3.4, along with the resulting effective A and Z (note, they are not used in the code, given just for orientation). The built-in default densities for each compound is also listed. The user can modify this density using the **MTDN**

card in the `MARS.INP` file as described in Section 4.2.3.

Table 3: Built-in compounds, with their default density values, and their average A and Z (these A and Z are not used by the code). The table starts with 10 elements in liquid phase with their densities.

Abbreviation	$\rho(g/cm^3)$	A	Z	Compound
LH	0.0708			Liquid hydrogen
LHE	0.125			Liquid helium
LN	0.807			Liquid nitrogen
LO	1.141			Liquid oxygen
LF	1.507			Liquid fluorine
LNE	1.204			Liquid neon
LCL	1.574			Liquid chlorine
LAR	1.396			Liquid argon
LKR	2.418			Liquid krypton
LXE	2.953			Liquid senon
GRPH	1.70			graphite
CH2	0.950	10.430	5.281	Polyethylene CH_2
BCH2	0.950	11.560	5.814	Borated Polyethylene
CH	1.032	11.159	5.613	Polystyrene
TEFL	2.200	17.320	8.280	Teflon, CF_2
NAI	3.670	110.97	46.56	Sodium Iodide crystal
ELEC	1.034	126.23	51.26	Electronics N-Si-Pb mixture
BGO	7.100	155.39	62.52	Bismuth germanate $(Bi_2O_3)_2(GeO_2)_3$
PBWO	8.280	170.87	68.36	$PbWO_4$
G10	1.700	17.164	8.583	NEMA G10 $60\%SiO_2 40\%epoxy$
QUAR	2.640	21.649	10.81	Quartz, SiO_2
TISS	1.000	13.798	6.952	Tissue, $H_{40}C_4NO_{17}$
WATR	1.000	14.322	7.217	Water, H_2O
PARA	0.930	10.376	5.257	Parafin wax $CH_3(CH_2)_{23}CH_3$
AIR	0.0012	14.803	7.374	Air at 18°C and 58% humidity
SOIL	1.900	20.485	10.21	Soil, $H_{17}O_{27}Al_2Si_9$
CONC	2.350	24.21	12.04	Concrete, with steel reinforcing, O2, Si, Ca, Na, Fe, Al
BART	3.200	74.16	31.33	Boron-Barite Concrete
BMCN	3.637	38.94	18.40	Borated Magnetite Concrete
STST	8.020	55.39	25.81	Stainless Steel 304
SCON	7.000	63.52	28.73	Super Conductor $90\%(60\%Cu+40\%NbTi)+10\%Kapton$
YOKE	7.870	55.81	25.99	Steel magnet yoke
COIL	5.108	62.89	28.70	Water-cooled copper coil
INSU	1.690	49.27	23.02	Insulation
CABL	2.896	48.78	22.52	Cables
CH4	0.4224	9.246	4.743	Methane CH_4
MYLR	1.390	12.88	6.46	Mylar $C_5H_4O_2$
MYAL	1.783	17.11	8.42	Aluminized Mylar
KAPT	1.420	12.70	6.36	Kapton $C_{22}H_{10}N_2O_5$
CRMC	3.970	21.81	10.65	Ceramic, Al_2O_3
TIAL	4.430	46.74	21.50	Ti-alloy, $0.9Ti+0.06Al+0.04V$ by weight
INCO	8.190	59.55	27.85	Inconel alloy 718
FECO	8.000	56.96	26.36	$0.64Fe+0.36Co$ by weight
NBSN	7.000	69.87	31.14	Nb3Sn SC coil
SCC	8.500	104.64	44.74	$0.24Nb3Sn+0.70CuSn+0.06Ta$
GFRP	5.682	45.63	20.86	$0.33STST+0.21CU+0.1SCC+0.2HE+0.16G10$
BITR	8.146	58.62	26.82	Magnet bitter coil, $0.9Cu+0.1water$
HOLW	5.50	43.56	20.21	Hollow conductor, $0.19water+0.25MgO+0.56Cu+9\%void$
DLMT	2.85	21.42	10.68	Dolomite with 15% water by volume
GTIL	2.90	27.05	13.13	Glacial till with 30% water by volume
SHTC	2.24	29.68	14.77	Shotcrete
NYLN	1.13	11.74	5.92	Nylon, Type 6, $NH(CH2)5CO = H11C6NO$
NDFB	8.00	63.75	28.82	NdFeB magnet, $0.06B+0.82Fe+0.12Nd$
VAC	0.	0.	0.	Vacuum

3.5   Tabulation *needs further update*

Results are accumulated and tallied in various zones in the modeled geometry. The zones can be volumes defined by the user's encoded geometry, or volumes and surfaces where histograms have been defined. A description of standard mars output information and how to use the results is given in Section 10.1 and Section 13.3.

Geometry and phase-space tagging options, used intensively in studies with MARS (see, e. g., [79, 80, 81]), have been further improved. These allow a very efficient way to study a source term, *and an example is given in Section 13.3.*

Calculated are three-dimensional distributions of star density, total and partial particle fluences above several thresholds, total and partial energy deposition densities, temperature rise, dose equivalent, residual dose rates, displacements per atom (DPA), one- and two-dimensional mass and charge nuclide distributions. All of these are accumulated and presented with corresponding statistical errors as part of the standard MARS outputs in the files MARS.OUT, MTUPLE, MTUPLE.NON, MTUPLE.EXG, MTUPLE.MCNP. Energy spectra in the defined zones, tagged distributions, and some integral values are also produced

3.6   Histogramming *needs further update*

The I/O sequence as well as the histograming for surface and volume detectors is substantially improved and extended.

MARS native and HBOOK-based histograming and analyses have been extended to include scoring of surface current, flux, dose equivalent, particle spectra and time distributions.

See XYZ-histograming section.

3.7   Visualization *needs further update*

See GUI Section.

3.8   Variance Reduction *needs further update*

Algorithms for splitting and Russian roulette at hA vertices (statistical weights) and in particle transport (particle trajectories). Phase-space and particle type biasing, exponential conversion of path length. Mathematical expectation: for 'deep penetration' problems in complex highly non-uniform geometries, algorithms for scoring probabilities rather than real particle crossings or interactions now take into account all possible processes for both stable and unstable particles and charged as well as neutral hadrons [17].

The user can now choose between sampling and forcing π-, K- and μ-decays. Algorithms for splitting and Russian roulette at hA vertices and in particle transport are also further improved. For 'deep penetration' problems in complex highly non-uniform geometries, algorithms for scoring probabilities, rather than real particle crossings or interactions, take into account all possible processes for both stable and unstable particles and charged as well as neutral hadrons. Use of accelerating field (RF-cavities) is now optional in the code.

3.9   Interfaces *needs further update*

Interfaces to the ANSYS code for thermal and stress analyses and to the STRUCT code for multi-turn particle tracking in accelerators have also been improved.

fusion with the STRUCT code [35] for multi-turn particle tracking in the accelerator lattice represented by an arbitrary combination of magnetic elements and transfer matricies allowing unified approach to beam loss and radiation effects studies at modern accelerators [82, 77, 83, 84, 85, 86];

3.10   Getting Started

This section presents an overview of the code and outlines how to create and run an executable. Details on the structure and content of the input deck are given in Section 4. Details on the usage of each of the user subroutines are found in Section 5. Descriptions of the various output files and examples of how to make use of them are given in Section 10 and Section 13.3.

MARS is primarily supported to run on SunOS and Linux operating systems, with secondary support SGI-IRIX, IBM-AIX, DEC-ALPHA and HP-UX. The MARS code system consists of a few hundred FORTRAN77 subroutines and a few CERN service *C*-routines, organized by functionality into several files. The source code file structure is listed in Table **??**. The computing efficiencies of many of the MARS software algorithms have been improved and optimized over time. The code runs completely in the IEEE-754 double precision model, [87], except for the CERN HBOOK package. MARS utilizes a machine independent universal random number generator [88].

The user subroutines, in file m1504.f, are the collection of routines which can be customized for a particular simulation. While these routines are FORTRAN, the user can of course write his routines in other languages, and have them called from the appropriate locations within the m1504.f routines. Input decks are also made available to the user for customization; the input decks consist of format-free, plain text data cards, which are read by the CERN FFREAD package. The output of MARS consists of several files, a few of which are always created and others which are created only when requested by the user via the input decks; many are text based tables, some are HBOOK files, and some are a specialized format for use as input in multi-step jobs.

The code is installed at many accelerator laboratories and universities. The code must be installed by the author, on request, and is not available for self-installation via anonymous ftp. For the latest information, a list of installation locations, code status, platform availability, comments and other related questions visit the official MARS Web site http://www-ap.fnal.gov/MARS or contact the author at mokhov@fnal.gov.

In general, the MARS code system is installed into a directory structure as shown below:

```
dat/          linux/    sun/          GEOM.INP
GNUmakefile   MARS.INP  XYZHIS.INP    m1504.f
marsmain.f    xsdir
```

The user has access to the source code only of the user subroutines in file m1504.f; all other source code is built into libraries. marsmain.f is a short file which holds the FORTRAN Program statement, the names of the main input and output files, and the call to the mars1504 master code:

```
      PROGRAM MARSMAIN
C.................................................
C     MARS15 CODE
C-----
C     LAST CHANGE: 04-AUG-2004 BY NVM
C-----
      IMPLICIT NONE

      INCLUDE 'fnames.inc'
C.................................................

      INPNAM          = 'MARS.INP'
      OUTNAM          = 'MARS.OUT'
      HBKNAM          = 'MARS.HBOOK'
      XYZTABNAM       = 'XYZHIS.TAB'
      DUMPNAM         = 'DUMP'
      MTUPLENAM       = 'MTUPLE'
      MTUPLEEXGNAM    = 'MTUPLE.EXG'
      MTUPLEMCNPNAM   = 'MTUPLE.MCNP'
      MTUPLENONNAM    = 'MTUPLE.NON'

      CALL MARS1504

      STOP
      END
```

The user is free to change the input/output file names in `marsmain.f`. The user copies marsmain.f, m1504.f, GNUmakefile, and the input decks MARS.INP, GEOM.INP, and XYZHIS.INP along with the MCNP's file `xsdir` to his/her own working area. The user customizes the input decks and the user subroutines as needed; the user is free to create additional subroutines called from the provided user subroutines. The GNUmakefile can be edited by the user to include the list of all the user's local subroutine files, so they will be compiled and linked to the MARS libraries to obtain an executable.

The executable is made by running the GNUmakefile script. This manual assumes the user is familiar with the syntax for compiling and linking source code using `gmake` or `make` on unix or linux systems. The GNUmakefile script in the installation directory uses compiler options appropriate for the code and for the operating system and compiler version on the machine where the code is installed. The GNUmakefile contains pointers to all the required external libraries, including those for graphics and for CERNLIB.

The directory sample above is located on a hypothetical cluster which contains various types of computer platforms. The directories named by platform type, linux/ sun/ irix/ hold the MARS object libraries compiled for that platform within a `lib/` subdirectory, and also hold a subdirectory of include files, `include/`, which contain MARS common blocks and variable declarations. On a single node or on a cluster of identical machines, there will be just one platform directory listed instead of the multiple platforms shown in this example.

Some geometric models which have a simple cylindrical structure can be described completely via the main MARS.INP input deck, using MARS Standard zones; an example of such a geometry is given in Section **??**. The sample MARS.INP file contained in the MARS installation directory is fairly simple and lists the items which most users will typically customize. It also lists, on the 2nd line of the file, the full path, on the user's local system, to the `dat/` files in the installation directory, which the MARS executable will need to load at run time. Other geometric models can be adequately described using groups of boxes, spheres and other standard shapes, and these models can be created

using MARS Extended Geometry zones via the GEOM.INP deck; an example of such a geometry is given in Section ref/inexextend. The sample GEOM.INP and MARS.INP.exgeom files are set up to illustrate the use of Extended zones (one must put INDX 3=T to MARS.INP to execute the example.

Both these simple geometric models, using only the MARS.INP file, or the MARS.INP and GEOM.INP files together, do not require customization of the user subroutines. In these cases, however, the user must still compile a local copy of the user subroutines to obtain an executable, even if the routines have not been modified from their initial dummy state.

For some problems involving the production and interactions of low energy neutrons, the user might consider using MARS coupled with the MCNP4 code library. See the discussion in Section 2.10 for the relevant conditions. The MCNP4 code is proprietary, and to utilize this coupled mode, the user must have a licensed mcnp4c2 version. More information on using MCNP4 with MARS is given in Section 6.

The MARS executable can be run interactively, in the background, or submitted to a batch system. By default, the executable and the input MARS.INP and GEOM.INP files must all reside in the same directory. For batch running, when the executable and input files might not be located in the same directory, the paths to the input files can be modified via customization of the user routine MARS1504, see Section 5.1 for more information. The running time per event can vary widely, depending on the complexity of the user's model. The user is advised to try short trial runs of 100 to 1000 events, to make an estimate of the cpu-time required. The more events which are generated, the better the statistical error on the results will be. In general, any zone of interest should have a relative error of no more than 20% for the results in that zone to have a physical validity. More discussion on this subject is in Section 13.1.

The interactive GUI interface is invoked by inserting a card, **CTRL 1**, in the main MARS.INP input deck. The interface is very useful to visually check the geometry of the model, before actually running the executable to generate events. This card must be set to **CTRL 0** to run MARS in the event generating mode, as this mode and the interactive GUI mode cannot be run simultaneously. Details on using the GUI interface are given in Section 11.

The default output files are MARS.OUT, MTUPLE, MTUPLE.EXG, MTUPLE.MCNP and MTU-PLE.NON; if histograms were requested there will also be a mars.hbook file. These file names can be changed in the file OPEN statements in the user subroutine MARS1504. MARS.OUT and MTUPLE list the results accumulated in MARS Standard zones, with some information on Non-Standard zones also contained within MARS.OUT. MTUPLE.EXG lists results accumulated in Extended Geometry zones. MTUPLE.NON lists results accumulated in Non-Standard zones. The format of these files is defined within non-user code, and so cannot be changed by the user. Details on the contents of these files, and other specialized output files, is found in Section 10.

4 Input Files *current with MARS15 - September 2004*

This section describes the various input files, or *decks*, used by MARS. These are the *native* input files, as opposed to user-created input files which might supplement the User Subroutines. The native input files are the main MARS Input Deck MARS.INP, the Extended Geometry input deck GEOM.INP and the XYZ Histograming input deck XYZHIS.INP. The MARS.INP file is used to define the main input parameters to be used in the run and the Standard Zones (minimum one mother volume). It also contains the MCNP section with a list of materials and (optionally) the MCNP geometry description used in MARS at IND(5) = IND(16) = T. The optional GEOM.INP file is used to define Extended Zones, and the optional XYZHIS.INP is used to define flexible XYZ-histograming.

Note again that different geometry descriptions in MARS – standard, non-standard, extended, MCNP and FLUKA – can co-exist and overlap in the same setup, in a single problem.

Besides that, numerous input files can be used by the user in a specific problem to provide information on special precalculated source terms (*e.g.*, DPM.EVE for event generators, BLOSS for beam loss distributions in the beam-line or accelerator lattice), magnetic field distributions in the detectors and accelerator elements (*e.g.*, QUAD1.MAP), particle distributions calculated at a previous stage for multi-stage runs, many optics and lattice element files used in the MAD-MARS Beam-Line Builder (MMBLB) mode, *etc*.

The main MARS.INP deck controls many features in MARS, such as : the primary beam parameters, materials, energy thresholds, termination conditions, hA-vertices parameters, scoring parameters, and regions where the standard histograms are accumulated, etc. Nearly all these features have default values built into the code; if the default values are appropriate for the user's model, then the control lines for those features do not need to be present in the MARS.INP file. Some problems cannot be described solely by information in MARS.INP, such as complex composite materials different of the built-in compounds, complex geometries, atypical primary beam distributions, or magnetic fields. In these cases, the appropriate user subroutines are customized to describe the problem's complexity, and the use of these is discussed Section 5. However, even with extensive code customization, the MARS.INP file parameters are a key method of specifying basic information.

Sections 4.1 and 4.2 cover all aspects of the main MARS.INP parameters. Section 4.3 describes the additional lines which get added to MARS.INP when using the MCNP libraries in conjunction with MARS. Section 13.2 describes in more detail the various 'switches' which turn on or off various physics process, and the conditions under which one might wish to control them. Section 4.4 describes the use of the GEOM.INP file. Section 4.5 describes the use of the XYZHIS.INP file. Section 4.6 holds examples which use only the input files to describe various geometries.

4.1 Structure of the MARS.INP Input Deck

The MARS.INP file, generally called the *input deck*, is always required to run a MARS program. At the very least, the MARS.INP file describes the geometry's Standard Zone(s), and the list of materials used in the geometry. The input deck consists of what are called cards, following ancient FORTRAN conventions; a card is a line or series of lines in the MARS.INP file. The name of the file is not required to be MARS.INP, however, if the user wishes to modify the file name, he must edit the corresponding line in the main program marsmain.f, Section 5.1.

The units used by the input deck parameters, and indeed used throughout MARS are: energy in GeV, dimensions in cm, angle ϕ in $degrees$, and magnetic fields in $Tesla$. The reference system is a

cartesian coordinate system, where the z axis is longitudinal, and the positive direction is from *left to right*. The z axis is usually the center of symmetry, and as a rule the primary particles strike along this axis in the positive direction. The positive direction of the x axis of the coordinate system is *up* and the y axis is *toward the viewer*, completing a right-handed system. Any other units used by parameters in the input deck will be specified in the list of input cards where appropriate.

The first two cards in the `MARS.INP` deck (lines in the file) are lines of text; the remaining are data cards. The order of the the data cards and their number is arbitrary. The only requirement is that the two text cards appear on the first two lines of the input deck, and that the **STOP** data card be present to terminate the sequence of data cards. Once in the MCNP mode (see Section 4.3), a block of lines with material and (optionally) geometry description is present after the **STOP** card. Any lines after that are ignored by the input deck initialization code.

The first line of the input deck is the Title card. It has a `FORTRAN` format of `A80`. The user can enter any text on this line, usually a short title to describe the problem, and perhaps the date, such as *Cu target in Area B with design 1-a shielding, March 15, 2004*. This line of text is read into character variable `MTEXT` in the code, and is written to the MARS.OUT file.

The second line of the input deck is the Path card, and also has a `FORTRAN` format of `A80`. The card gives the full directory path to the location of the `dat/` directory in the `MARS` installation directory, and as such is case sensitive. Example: */home/mokhov/restricted/mars15/dat*. This line is read into character variable `DIRECT` in the code, and is used in `OPEN` statements to read the files of cross-sections and other data which `MARS` requires. The sample input deck MARS.INP located in the `MARS` installation directory will have the correct full path for that system. If the path is incorrect, a message will be written to the screen file stating that the required data cannot be located.

The rest of the MARS.INP file consists of data cards. The CERN `FFREAD` package is used to read these data cards in the routine `BEGINN`. The cards are format-free in the sense that they can appear in any order, and the qualifiers within each card can be listed in any order. However, there are structural conventions which must be followed for the cards to function properly.

The structure of all these cards is the same:

KEYW `a(1) a(2)...` N_{b1}`=b(1) b(2)...` N_{c4}`=c(4) c(5) ...`

KEYW is a keyword assigned to a group of variables `a(i)`, `b(j)` and `c(k)`. The variables are mapped onto a one dimensional array `A(i+j+k)`. Within array `A`, N_{b1} is the location where element `b(1)` is located; N_{c4} is where element `c(4)` is located. The numerical values listed after **KEYW** will be loaded sequentially into array `A`, until a "$N=$" is encountered. Once detected, the filling of array `A` will jump to the location given by N, and continue filling sequentially from there.

For example, there is a data card with keyword **ENRG**, which controls the primary energy and energy threshold cutoffs, below which particles will not be tracked. **ENRG** is assigned to nine variables. All of these thresholds have default values. For this example, all of the defaults are satisfactory for a given model, except the photon energy cutoff, which the user would like to raise from the default 0.0002 GeV to 0.001 GeV. That variable, `EMIGA`, is the 6th of 9 listed. The user would enter

ENRG $6 = 0.001$

into the input deck. This directive will change the default value of the 6th variable to 0.00 GeV.

If the user next wished to also change the default value on the 4th variable (`EMCHR`, the cuttoff on charged hadrons and muons), then this card entry would become

ENRG $4 = 0.03$ $6 = 0.001$

A space must separate the two equality statements, but that is all that is required.

Finally, if the user decided to additionally change the default on the 5th variable (`EMNEU`, the cut-toff on neutrons), then the card becomes

ENRG $4 = 0.03$ $5 = 0.05$ $6 = 0.001$

which is equivalent to

ENRG $4 = 0.03$ 0.05 0.001

For the previous example all the variables were of dimension (1). The example would appear the same if there was a single variable of dimension (9) assigned to **ENRG**. An example of cards which are assigned multiple variables with dimensions > 1 is given in Section 4.6.

The variables associated with a card can be integer, real, logical (represented by `T` or `F`), and, in one case, character. In general, all the variables assigned to a card are of the same type, but there are a few cards of mixed integer and real variables. The numerical values for real variables must contain a decimal point, for example "700" is not correct for a real variable while "700." is. The message will be generated to the screen if a real number value is missing the decimal point.

The card is ignored and treated as a comment if it starts with **C** separated by a space from the rest of the card.

4.2   List of Data Cards in MARS.INP

The listing of Data Cards is organized in functional groups: cards used in the overall control of the program; cards used to describe the materials and material dependent parameters; cards used to describe the Standard Zone geometry; cards which control histogramming and tabulation; and cards to control specific physics interactions. Cards are generally listed alphabetically within these functional groups.

4.2.1   Overall Control

CTRL `IVIS IEVT IVOL IHIS`

> Integer variables which control alternative running modes: the normal Monte Carlo session, the use of the GUI visualization, the size of the GUI window, the use of the event generator in "stand-alone" mode, Monte Carlo calculation of volumes for arbitrary regions, and a histogram palette in the GUI visualization. Each of these, along with the normal running mode, are independent of each other and cannot be used concurrently. Details on using the GUI visualization interface are in Section 11, and details on using the event generator in stand-alone mode are in Section 5.1.

> `IVIS`   When `IVIS=0`, MARS will run in it's normal simulation mode. If `IVIS`$\geq$`1` then the MARS GUI interface is activated. No events will be generated, and the visualization interface screen will pop up when the executable is run. The value of `IVIS` determines the size of GUI window: `1` gives a 15-inch window, `2` gives a 14-inch window, and `3` gives a 13-inch window. Default: 0.

IEVT   If `IEVT`$\geq$`1`, then regardless of the value of `IVIS`, MARS will run in it's stand-alone event generator mode. The primary particle energy and type are specified using the **ENRG** and **IPIB** cards, interacting on nucleus type `IEVT=IM`, where `IM` is the index of the selected material as listed in the **MATR** card. The generator is run for `NSTOP` events as given in the **NEVT** card. Default: 0.

IVOL   If `IVOL=1`, then regardless of the value of `IVIS` and `IEVT`, MARS will run a short Monte Carlo session to calculate volumes for non-standard or any other (overlapped) regions contained in the cylinder defined by the the **RZVL** card. The results go into an output file VOLMC.NON in the format ready for inclusion into the user routine VFAN. Statistical errors for the calculated volumes are also included to see if a number of events `NVTRIAL` in this session is enough. Default: 0.

IHIS   When `IHIS=0`, a detailed 16-color palette is used for histogram presentation in the GUI, optimized for color view of results calculated spaning up to 16 decades. If `IHIS=1`, then a 8-color palette is used for histogram presentation, convinient for figures to be included in a black and white paper. Default: 0.

RZVL `R1 R2 Z1 Z2 NVTRIAL`

Variables which define a cylinder containing regions (non-standard, MCNP, extended or overlaped), which volumes a user wants to calculate in a short Monte Carlo session. That is required for the code to normalize result per unit volume or unit mass. By default this cylinder is the mother volume, but one can define an arbitrary smaller cylinder inside the mother volume to get better statistics for specific regions, especially for tiny ones. If the system studied is bulky and contains many tiny non-standard objects at large distances from each other, it is recommended to run several short jobs defining the cylinders surrounding the region groups of interest. Statistical RMS errors in volume calculation is printed in the VOLMC.NON file and can be reduced by increasing `NVTRIAL`. Typically it takes a few tens of seconds on a typical workstation to get the above errors $\leq$ 1% with `NVTRIAL` of the order of a few millions.

R1, R2, Z1, Z2   Real variables for minimal and maximal radii and z-coordinates of a cylinder which contains the non-standard regions whose volumes need to be calculated. Default: 0., `RMAX`, `ZMIN`, `ZMAX`.

NVTRIAL   Integer or real variable equal to the number of events to run in this volume determination session. Default: 10^7.

INDX `IND(16)`

Logical variables which control various options, described by the list below. Default: `9*F 10=T 6*F`.

IND(1)=T   provides the maximum amount of output to be printed and time-dependent residual dose in all materials. See Section 10.1.

IND(1)=F   provides a reduced output with no details on residual dose. See Section 10.1.

IND(2)=T   initiates "$Z - sandwich$" geometry as a basis for the Standard Geometry zones. (see `ZSEC` and `RSEC`).

IND(2)=F   initiates "$R - sandwich$" geometry as a basis for the Standard Geometry zones. (see `ZSEC` and `RSEC`).

IND(3)=T   activates the Extended Geometry option, where a model is described using combinations of boxes, cylinders, cones, etc. See Section 4.4 for a description for using the GEOM.INP input deck, and Section 4.6.4.

IND(3)=F   Extended Geometry option turned off.

IND(4)=T   magnetic or electric fields are present in the geometry description. This flag triggers
            calls to the user subroutine FIELD, where field components must be defined.

IND(4)=F   no magnetic or electric fields.

IND(5)=T   initiates the use of the MCNP4C–ENDF/B-VI system for neutron transport below
            14.5 MeV, with corresponding photon production.

IND(5)=F   use the default neutron transport in the 0.00215 eV to 14.5 MeV energy range using
            the 28-group neutron cross-section library with no low-energy photon treatment.

IND(6)=T   use a mathematical expectation method to accumulate the results for transported
            particles: probabilities are used to tabulate fluence, star density, particle spectra and
            the calculation for the dose equivalent, rather than directly tabulating the particle
            contributions (analog method). This option is useful for any "deep penetration"
            problems, i.e. thick shields; however, it can substantially increase the CPU-time,
            especially when neutron cutoff energy is below a few MeV. It is recommended to
            conduct a short test to evaluate an efficiency of this option for a given application.
            See Section 13.1 more more discussion. Note that direct energy deposition calcula-
            tion and related values are not affected by this flag.

IND(6)=F   analog method for tabulation of results of transported particles; appropriate for ge-
            ometries which do not model thick shielding.

IND(7)=T   indicates that an incident particle interacts, with a certain probability, with the point-
            like target placed in the system at coordinates (x_0, y_0, z_0). The probability is spec-
            ified by the EFF variable associated with the **VARS** data card. By convention the
            material index for this target is IM=1, but this can be re-defined in the BEG1 user
            subroutine. This also initiates forced μ-decay at $I0$=7 or 8 without ν-transport.

IND(7)=F   no point-like target or forced μ-decay at $I0$=7 or 8.

IND(8)=T   neutrino beam and transport (both primary and from π and μ-decays).

IND(8)=F   no neutrino beam and transport.

IND(9)=T   initiates the use of special algorithms (as in [30, 89]) to construct extremely accu-
            rately the particle trajectories prior to the first two inelastic nuclear interactions. A
            default algorithm in the current code provides the same level of accuracy, therefore
            this options practically obsolete.

IND(9)=F   does not use the above algorithms.

IND(10)=T   Default. The flag activates muon transport with all possible interaction processes
            included (see Section **??**).

IND(10)=F   turns off muon transport.

IND(11)=T   adds the $\phi-$ dimension to the Standard (*r-z*) Geometry zones, giving an azimuthal
            structure to the accumulation of the results. The user must specify the number and
            size of the azimuthal zones via the NAZM and AZIM data cards.

IND(11)=F   no azimuthal divisions in the Standard Geometry zones.

IND(12)   reserved for the point-kernel option to calculate residual radioactivity with routine
            MARACT; not implemented in this version.

IND(13)=T   The flag activates the MAD-MARS beam line builder (MMBLB) interfaced with the
            MAD lattice description along with histogramming in bend beam-lines and acceler-
            ators using user subroutines MARS2BML and BML2MARS.

IND(13)=F   MMBLB is not activated.

IND(14)=T   User subroutines ALIGN, SAGIT and RFCAVT are called by the MARS geometry
            description software modules.

IND(14)=F   The above subroutines are not called.

IND(15)=T   Enables a "global parameter set for thick shielding". The set consists of various physics options, energy thresholds, and tabulation options which are collectively appropriate when modeling thick shielding. Can be used with IND(6)=T. Setting this flag can provide a *substantial* CPU-time saving for models of thick extended shielding. The particular parameters and their settings are given in the table below. As with all similar global settings, any of the individual parameters in this set can be re-set, globally or for specific materials, using the appropriate input deck parameter

IND(15)=F   The above items in the "global parameter set for thick shielding" are set to standard MARS defaults

IND(16)=T   MCNP code geometry is used decribed in the MCNP section of the MARS.INP file at IND(5)=T.

IND(16)=F   MCNP code geometry is not used.

	IND(15)=F (default)	IND(15)=T
1. Charged hadron multiple Coulomb scattering	Always	First three generations of cascade tree only
2. Charged hadron nuclear elastic scattering at $E > 5$ GeV	Always	First three generations of cascade tree only
3. Knock-on electron production by hadrons with detailed modeling of induced electromagnetic showers	Always	First three generations of cascade tree, continuous dE/dx after
4. Knock-on electron production by muons with detailed modeling of induced electromagnetic showers	Always	For muon energies $E/E0 > 0.2$, continuous dE/dx after
5. Direct e^+e^- pair production by muons with detailed modeling of induced electromagnetic showers	Always	For muon energies $E > 5$ GeV and $E/E0 > 0.2$, continuous dE/dx after
6. Global cutoff energies:		
Charged particles	0.2 MeV	Maximum of 3 MeV and user-defined
Photons	0.2 MeV	Maximum of 1 MeV and user-defined
Neutrons	0.1 MeV(*)	Maximum of 0.1 MeV and user-defined
7. Boundary localization precision STEPEM (first parameter on SMIN card)	User-defined, 0.01 cm typically	Maximum of 0.3 cm and user-defined
(*) Used to be 10^{-12} GeV in previous versions.		

NEVT NSTOP NTIME NHIPR IPRINM

Integer variables which control the number of events to run and print, and control printout of a given history and of nuclear cross-sections.

NSTOP   the number of events to generate (equals incident particles thrown). Default: 200.

NTIME   the number of times user routine DUMP will be called. Each call writes intermediate results to the DUMP file. Recommended: 10. Default: 0.

NHIPR   the event number which will be written in great detail to a file `fort.30`. If `NHIPR=-1` then the number of the current event being generated is printed to the screen (stout). Default: 0.

IPRINM   If `IPRINM=1` then detailed tables of hadron and photon nuclear cross-sections being printed for each material as a function of energy. Default: 0.

SEED `IJKLIN NTOTIN NTOT2N`

Integer variables to initialize the random number generator, and to track how many random numbers are used.

IJKLIN   The initial 64-bit seed. The integer entered is assumed to be in octal, therefore no digit should be greater than 7. Default:54217137; shift example:64217136.

NTOTIN   Default: 0

NTOT2N   Default: 0

SMIN `STEPEM STEPH`

Real variables specifying global boundary localization precision and pilot step lengths. The program tracks particles in line segment steps (see discussion in Section 3.3. The pilot step, `STEPH` is used initially if it is smaller than that defined from physics and magnetic field criteria; if within that step the zone number changes, then the program will iterate back and forth, in smaller and smaller steps, to locate the boundary between the zones (the boundary is located exactly without iterations in the Extended Geometry mode). The minimum size of these iterative steps is given by `STEPEM`. The larger the step lengths, the faster the code will run, but accuracy is affected if the step lengths are so large that smaller size zones become "invisible". One can, however, specify fairly large (or small) step lengths here, and then apply smaller (or larger) step lengths to particular materials via variables assigned to the **MTSM** and **MTSH** cards. See the example in Section 4.6.3.

STEPEM   The global parameter, in cm, for the accuracy of boundary localization in the particle transport algorithm. It is strongly recommended to define `STEPEM` to be no larger than $0.1 \times t_{min}$, where t_{min} is the smallest dimension of the smallest zone in the model. Default: 0.01

STEPH   The global pilot step length, in cm. While making the value large has a modest effect on the time per event ($\simeq \log(STEPH/STEPEM)$. The recommendation is to set `STEPH`$= \min(\lambda, l)$, where λ is the mean inelastic length for hadrons and l is the length of the zones of interest, in the direction which most particles pass through those zones. Default: 10 cm.

VARS `EFF DLEXP TEMPO AINT`

Real variables which control miscellaneous aspects of the problem.

EFF   The point-like target efficiency, active only when `IND(7)=T`, $0 \leq$ `EFF` ≤ 1. Default: 0.

DLEXP   An exponential conversion factor, which allows an increase (`DLEXP`≥ 1) or decrease (`DLEXP`≤ 1) in the effective inelastic mean free path. This feature is useful where the "deep penetration" problem applies, and for compact restricted models. Recommendation: $0.3 \leq$ `DLEXP` ≤ 3. Default: 1.

TEMPO   The initial temperature T_0, in $Kelvin$, of the model (all zones). The allowed range is $1.8 \leq T_0 \leq 1800$. Default: 300.

AINT   The number of particles, N_0, in a given pulse of beam used for two non-linear values. The format is 4=1.0e12. The assumed units for the value on what output information the user requires. If the user needs to understand an instantaneous temperature rise due to beam, then N_0 is assumed to be the number of particles in a single instantaneous pulse. If the user is more interested in an estimate of the contact residual dose, power density, radiation damage (DPA per year), then N_0 is assumed to be an average beam intensity in particles per second. Default: 10^{12}.

4.2.2   Primary Beam

BEAM SIXX SIYY SITX SITY DLBNCH

Real variables used to specify the x, y, z beam shape. The first four variables are active only for non-zero values of IBEAM in the **IPIB** card. The default for each is 0.0.

SIXX   For IBEAM=1, the half-size in x of a uniform retangular beam; for IBEAM=2,3, the σ_x of a Gaussian beam. Both in cm.

SIYY   For IBEAM=1, the half-size in y of a uniform retangular beam; for IBEAM=2,3, the σ_y of a Gaussian beam. Both in cm.

SITX   The gaussian angular spread $\sigma(\theta_x)$, in radians; used only when IBEAM=3.

SITY   The gaussian angular spread $\sigma(\theta_y)$, in radians; used only when IBEAM=3.

DLBNCH   The RMS bunch length σ_z of a Gaussian beam in cm (30 cm = 1 ns).

ENRG E0 EM EPSTAM EMCHR EMNEU EMIGA EMIEL EMNU EMEVAP

Real variables which control the incident particle energy, and energy thresholds for subsequent generated particles. In most cases, particles will not be tracked once their energy is below the threshold, and their contribution to flux and energy distributions will no longer be counted; exceptions noted below. These thresholds are global; material specific thresholds can be specified using variables assigned to the **MTCH**, **MTNE**, **MTGA**, and **MTEL** data cards.

E0   The incident particle kinetic energy. Default: 100.0 GeV.

EM   The hadron threshold energy, but only for flux and spectra tabulation, and not for energy deposition. Default: 0.0145 GeV. However, if this value is greater than the default 0.0145 GeV, then it becomes a global threshold for the transport and tabulation of **ALL** particles and not just hadrons. In that case, the results for energy deposition will be substantially underestimated, but the CPU-time will have decreased x10. For example, in problems where star density is the main result of interest, the user should set EM to 0.05 GeV.

EPSTAM   The star production threshold kinetic energy. Default: 0.05 GeV.

EMCHR   The global threshold energy both for charged hadrons and for muons. Default: 0.0002 GeV.

EMNEU   The global threshold energy for neutrons. Default: 10^{-4} GeV, used to be 10^{-12} GeV in previous versions.

EMIGA   The global threshold energy for γ. Default: 0.0002 GeV.

EMIEL   The global threshold energy for e. Default: 0.0002 GeV.

EMNU   neutrino energy threshold. Default: 0.01 GeV.

EMEVAP   The minimal energy for pre-equilibrium and evaporation modeling, which is turned off if EMEVAP$\geq$0.2 GeV. Affects nucleons only. Can substantially reduce CPU in high-Z materails. For example, useful if one is interested in π/μ production only. Default: 0 GeV.

INIT `XINI YINI ZINI DXIN DYIN DZIN WINIT`

Real variables used to specify the initial starting point and direction cosines of the incident beam spot center. Complex initial beam directions can also be modeled, using these same variables, via the user subroutine `BEG1`.

`XINI,YINI,ZINI`   These are initial x_0, y_0, z_0 coordinates of the beam spot center. Default: $0.0, 0.0, 0.0$, in cm.

`DXIN,DYIN,DZIN`   These are initial direction cosines of the beam centroid. Default: $0.0, 0.0, 1.0$

`WINIT`   The initial weight of each incident particle, which the results are normalized to. Watch for interference with `AINT`. Default: 1.0.

IPIB `I0 IBEAM IZPRJ IAPRJ`

Integer variables to specify the incident particle type, the incident beam distribution and the charge and mass for incident heavy ions. If a non-zero `IBEAM` value is given, then the parameters of the profile are defined using the **BEAM** card; in this case the applicable profile parameters must be non-zero. More complex beam profiles must be modeled via the user subroutine `BEG1`.

`I0`   The incident particle type; see Table 3.2 for the index values assigned to various particles. Default: 1, proton.

`IBEAM`   The type of incident beam distribution, given by one of the four following values. Default: 0.
-1 – laterally infinitesimal beam
0 – "a diffused track": the beam is distributed uniformly in rectangular areas $\pm 10^{-5}$ cm around `XINI, YINI, ZINI`, default.
1 – the beam is distributed uniformly in a rectangular area with half-sizes specified by **BEAM** card variables.
2 – beam is a simple Gaussian with σ_x and σ_y specified by **BEAM** card variables.
3 – beam has a Gaussian spatial distribution, as for `IBEAM=2`, and also has a Gaussian angular spread with $\sigma(\theta_x)$ and $\sigma(\theta_y)$, in radians, which are specified using the **BEAM** card.

`IZPRJ`   Integer electric charge of the projectile if it is heavier than ^4He. Default: 1.

`IAPRJ`   Integer mass (baryon) number of the projectile if it is heavier than ^4He. Default: 1.

4.2.3 Materials

NMAT `NREMA`

Integer variable specifying the number of materials in the model. This value defines to the program the number of active array elements in the variables assigned to the **MATR**, **MTDN**, **MTCH**, **MTNE**, **MTEM**, **MTSM**, **MTSH** data cards. The one special material, *blackhole* is not included in the total number. The program assigns `IM=` -1 and region number $N \leq 0$ to *blackholes*, which are either all material outside of the simulated volume ($N = 0$) or any region inside marked with $N = -1, -2, -3, ...$ where entering marticle is

immediately killed. Usefull for special scoring or forming a source term for the cosequent runs, both in subroutine LEAK. Another special material, *vacuum* can be used either the same way, i.e. not included in the total number, the program assigns IM= 0 (usually it is a global vacuum), or as any other material listed using (multiply if needed) the name VAC with corresponding IM> 0. Last option is useful if one needs to assign different transport properties (step parameters and cutoff energies) to the different vacuum regions.

NREMA   The total number of different materials used in the model. 1≤NREMA≤100. Default: 1.

MATR AMA(100)

Character variables which specify the names of the NREMA materials in the model. Each of these materials has an index, IM= $1 : NREMA$, following the order in which they are listed on this card. The material-dependent density, energy thresholds, and step lengths in the **MTDN**, **MTCH**, **MTNE**, **MTEM**, **MTSM**, **MTSH** data cards use the same IM index.

Materials can be single elements or complex compounds. Section **??**, lists all the 153 built-in materials, which includes elements up to atomic number 100, and 53 of the most common compounds used in the accelerator/detector/shielding world.

The user can define his/her own compounds with names MIX1, MIX2, MIX3, etc., up to MIX0 for the first 10, and continue using the names MIXA, MIXB, MIXC ... for subsequent mixtures. Each of these is assigned an IM index, as above, according to their order in the list of materials. The user defines each mixture, identified by the IM index, in subroutine MIXTUR (see Section 5.2).

A particular material can be listed more than once, if the user wishes to specify different properties to be applied to that material when it occurs in different regions of the geometry. For example, a model might have both steel shielding and thin steel vacuum pipe. For the steel shielding, a large STEPEM and STEPH could be specified, using the global step control parameters assigned to the **SMIN** card. The thin steel pipe could have a smaller step parameters applied, by using the **MTSM** and **MTSH** cards. The material FE would be listed twice so that two IM indicies are obtained. The shielding zone and the pipe zone would have these different material indicies assigned, even though they are both steel, so that the appropriate step length properties get applied by the code. See the example in Section 4.6.3.

AMA   The material name, in the syntax exactly as shown in Tables 3.4 and 3.4 : 'XXXX', including the single quotes, where XXXX is the 2 to 4 character abbreviation for the material. Default: 'FE'.

MTDN ROW(100)

Real variables giving the density, in g/cm^3, of the materials listed in the **MATR** card. All of the elements listed in Table 3.4 have their standard density value built in. All of the compounds listed in Table 3.4 have typical density values built in (usually from PDG). If the user wants to apply a different density value, for example if his graphite target has a density different from standard, then this card is used. User-defined mixtures are required to have their density specified by using this card. The default material is FE, and it's standard density is the default value, which means that for the default case, this card does not need to be listed in the input deck. If the density specifed $\geq 10^{20}$ g/cm^3 then this material is treated as a *blackhole*. If the density specifed $\leq 10^{-20}$ g/cm^3 then this material is treated as a *vacuum*. In both case a corresponding message will be written to the screen.

ROW   The values for the material densities, indexed by `IM`, the order the materials are listed in the **MATR** card.

MTCH `RLCTCH(100)`

Real variables giving the charged hadron and muon energy threshold applied only to specific materials. Any material which does not have a corresponding entry here will have the global threshold applied as the default, with the value given by the `EMCHR` variable in the **ENRG** data card.

RLCTCH   The values for the energy thresholds, indexed by `IM`, the order the materials are listed in the **MATR** card.

MTNE `RLCTNE(100)`

Real variables giving the neutron energy threshold applied only to specific materials. Any material which does not have a corresponding entry here will have the global threshold applied as the default, with the value given by the `EMNEU` variable in the **ENRG** data card.

RLCTNE   The values for the energy thresholds, indexed by `IM`, the order the materials are listed in the **MATR** card.

MTGA `RLCTGA(100)`

Real variables giving the electromagnetic energy threshold, for γ, applied only to specific materials. Any material which does not have a corresponding entry here will have the global threshold applied as the default, with the value given by the `EMIGA` variable in the **ENRG** data card.

RLCTEM   The values for the energy thresholds, indexed by `IM`, the order the materials are listed in the **MATR** card.

MTEL `RLCTEL(100)`

Real variables giving the electromagnetic energy threshold, for $e^{\pm}$, applied only to specific materials. Any material which does not have a corresponding entry here will have the global threshold applied as the default, with the value given by the `EMIEL` variable in the **ENRG** data card.

RLCTEM   The values for the energy thresholds, indexed by `IM`, the order the materials are listed in the **MATR** card.

MTSM `RLSTEM(100)`

Real variables giving the step length for boundry localization, applied only to specific materials. Any material which does not have a corresponding entry here will have the global step length applied as the default, with the value given by the `STEPEM` variable in the **SMIN** data card. The recommendation for the value of this parameter is the same as described for the `STEPEM` variable.

RLSTEM   The values for the step lengths, indexed by `IM`, the order the materials are listed in the **MATR** card.

MTSH `RLSTEH(100)`

Real variables giving the pilot step length, applied only to specific materials. Any material which does not have a corresponding entry here will have the global pilot step length applied as the default, with the value given by the `STEPH` variable in the **SMIN** data card. The recommendation for the value of this parameter is the same as described for the `STEPH` variable.

RLSTEM   The values for the step lengths, indexed by `IM`, the order the materials are listed in the **MATR** card.

4.2.4   Standard Geometry or Mother Volume

ZMIN `ZLEFT`

Real variable giving the minimum (left-most) value for the Z-coordinate. The value can be negative if necessary. The starting location for the primaries, `ZINI` in the **INIT** data card, may be moved to match this coordinate, or placed at a larger coordinate; this depends on the details of the user's model.

ZLEFT   The left-most Z-coordinate. Default: 0.

NLNG `LZ NLZ`

Integer variables which define the number of major longitudinal divisions of the MARS Standard geometry zones. See Section 4.6.1 for further discussion.

LZ   The number of major longitudinal sections described in the accompanying **ZSEC** data card. The allowed range is $1 \leq LZ \leq 1250$. Default: 1

NLZ   The number of times the major sections described in the **ZSEC** card are repeated. Active only for $NLZ \geq 2$. Default: 1

ZSEC `ZSE(1250) IZN(1250) IZI(1250)`

Real and integer variables which define the locations of the major longitudinal divisions of the MARS Standard geometry zones and the number of sub-sections within each. The materials in those zones are also specified by this card, when `IND(2)=T`, which is for a $Z - sandwich$ type geometry; see Section 4.6.1 for further discussion. Recall from the discussion in Section 4.1 that the three variables of dimension 1250 are mapped onto a single array of dimension 3750.

The boundary divisions are contiguous (no gaps) and must be listed in ascending order. The maximum Z coordinate in the model, `ZMAX`, is the right-hand boundary of the last declared division, or the last `ZSE` value listed.

ZSE(i)   The z-coordinate (real number) of the right-hand boundary of the ith longitudinal section. `ZLEFT` in the **ZMIN** data card gives the left-hand coordinate of the 1st section. These values occupy elements 1:1250 of the mapped array. Default: 100., 1249*0.

IZN(j)   The integer number of minor subsections within a given major section. Elements `j=1251:2500` of the mapped array correspond to the `i=1:1250` major sections. Default: 1250*1

IZI(k)   The material index value `IM` of each major section. The minor subsections must be of the same material as their major section. Elements `k=2501:3750` of the mapped array correspond to the `i=1:1250` major sections. This variable is active only when `IND(2)=T`. Default: 1250*0

41

NLTR `LR`

Integer variable which defines the number of major radial divisions of the MARS Standard geometry zones. See Section 4.6.2 for further discussion.

`LR` The number of major radial sections described in the accompanying **RSEC** data card.. The allowed range is $1 \leq \texttt{LR} \leq 20$. Default: 1

RSEC `RSE(50) IRN(50) IRI(50)`

Real and integer variables which define the locations of the major radial divisions of the MARS Standard geometry zones and the number of sub-sections within each. The materials in those zones are also specified by this card, when `IND(2)=F`, which is for a $R - sandwich$ type geometry; see Section 4.6.2 for further discussion. Recall from the discussion in Section 4.1 that the three variables of dimension 50 are mapped onto a single array of dimension 150.

The boundary divisions are contiguous (no gaps) and must be listed in ascending order. The maximum R coordinate in the model, `RMAX`, is the outer boundary of the last declared division, or the last `RSE` value listed.

`RSE(i)` The r-coordinate (real number) of the outer boundary of the ith radial section. The inner coordinate of the 1st section is assumed to be 0. These values occupy elements 1:50 of the mapped array. Default: 5.,49*0.

`IRN(j)` The integer number of minor subsections within a given major section. Elements `j=51:100` of the mapped array correspond to the `i=1:50` major sections. Default: 50*1

`IRI(k)` The material index value `IM` of each major radial section. The minor subsections must be of the same material as their major section. Elements `k=101:150` of the mapped array correspond to the `i=1:50` major sections. This variable is active only when `IND(2)=F`. Default: 50*0

NAZM `NF`

Integer variable which defines the number of azimuthal divisions of the MARS Standard geometry zones. This card is valid only when `IND(11)=T`.

`NF` The number of azimuthal bins, $1 \leq \texttt{NF} \leq 60$. Default: 1

AZIM `FIB(60)`

Real variables which give the angular size, in degrees, of each azimuthal division. This card is valid only when `IND(11)=T` and $\texttt{NF} \geq 2$.

`FIB` The angular sizes, $0 \leq \texttt{FIB(i)} \leq 360$. Default: 60*0.0

IRFL `IRFL(3)`

Integer variables which control the reflection of the geometry described for the positive x, y, z-directions to the negative ones. If `IRFL(1)=1` or 2, the geometry description at $x \geq 0$ is reflected to the negative x, with volume doubling at `IRFL(1)=1` and without volume doubling at `IRFL(1)=2`. Similar reflections are done independently for y and z directions at `IRFL(2)=1` or 2 `IRFL(3)=1` or 2, respectively. In the current version the reflection(s) have the same region numbers as the ones on the positive sides. Therefore the new region volumes are increased by a factor of 2 for each reflection. The resulting region volume increase factor `MVIRFL` is equal to 2, 4 or 8 depending on the number of

reflections. It is printed to the screen and output files and is used to calculate values in MARS.OUT and MTUPLE* which are normalized per volume or mass. Histograming is not affected by reflections because it is independent of geometry. For ABS(z) histograming use MHIRFL=1 on `NOBL` card of RZ-histograms.

IRFL   The reflection flags (0, 1 or 2). Default: 3*0.

4.2.5 Importance Sampling

IMPT `EIMPT NIMPTZ NIMPTR`

Real and integer variables which define the threshold energy for hadron importance sampling and the numbers of surfaces for that. See Section **??**. The surfaces and importances are defined in the accompanying **IMPZ** and **IMPR** data cards. Put these surfaces at large distances from the shower core to attack a deep penetration problem, never in vacuum or at a boundary with vacuum.

EIMPT   The cutoff energy for hadron weight splitting and Russian roulette. Default: 10^{-4} GeV

NIMPTZ   The number of the z-planes for hadron weight splitting and Russian roulette, $1{\leq}$`NIMPTZ`${\leq}10$. Default: 0

NIMPTR   The number of the cylindrical surfaces for hadron weight splitting and Russian roulette, $1{\leq}$`NIMPTR`${\leq}10$. Default: 0

IMPZ `ZZIMPT(i),WZIMPT(i),i=1,NIMPTZ`

Real variables which define the importance sampling in the z-direction.

ZZIMPT z-coordinates which define the planes for hadron weight splitting and Russian roulette. Place them in the exponential attenuation region well beyond the shower maximum. $z_1 < z_2 < ... < z_{10}$. Recommended $\Delta z \approx \lambda_{in}$, where λ_{in} is a material-dependent inelastic nuclear mean free path. In typical thick shielding case use it for neutrons of energy of several hundreds MeV. Default: 10*0.0

WZIMPT   Importance factors at the above z-planes which define the factors for hadron weight splitting and Russian roulette. If $\Delta z \approx \lambda_{in}$, then recommended $WZIMPT(i) \approx e \approx 2.7$. Default: 2.0

IMPR `RRIMPT(i),WRIMPT(i),i=1,NIMPTR`

Real variables which define the importance sampling in the radial direction.

RRIMPT   Radii which define the cylindrical surfaces for hadron weight splitting and Russian roulette. Place them in the exponential attenuation region at radii exceeding several λ_{in}. $r_1 < r_2 < ... < r_{10}$. Recommended $\Delta r \approx \lambda_{in}$. Default: 10*0.0

WRIMPT   Importance factors at the above radii which define the factors for hadron weight splitting and Russian roulette. If $\Delta r \approx \lambda_{in}$, then recommended $WRIMPT(i) \approx e \approx 2.7$. Default: 2.0

4.2.6 Histograms and Tabulation

NOBL `NOB MHIRFL NHSPE`

Integer variables which give the number of special RZ-regions in which all "volume type" histograms will be accumulated, the flag that controls z-histograming, and the type of scale for spectral histograms. See Section 10.3.1 for the list of the histograms and their IDs. The size of the volume of each region is given in the accompanying **RZOB** data card. If MHIRFL=1 then the ABS(z) histograming is activated, i.e., the code collects negative z-hits to histograms defined at $z \geq 0$, with an appropriate normalization. This is especially useful in a geometry reflection mode activated by a IRFL card.

NOB   The number of regions. $0 \leq$NOB≤ 3. Default: 0

MHIRFL   Activates ABS(z) histograming if MHIRFL=1. Default: 0

NHSPE   The type of scale on the particle energy spectra histograms: dN/dE if NHSPE=0 and $dN/d\log_{10}E$ if NHSPE=1. Default: 0

RZOB RZO(4,3)

Real values which specify the location and size of each special region. This card is valid only if NOB in the **NOBL** card is ≥ 1. See Section 4.6 for an example.

RZO(1,i)   The minimum radius of the ith special region, RMI(i), where $0 \leq$RMI$\leq$RMAX. Default: 0.0

RZO(2,i)   The maximum radius of the ith special region, RMA(i), where RMI$\leq$RMA$\leq$RMAX. Default: 0.0

RZO(3,i)   The minimum (left-hand) z-coordinate of the ith special region, ZMI(i), where ZMIN$\leq$ZMI$\leq$ZMAX. Default: 0.0

RZO(4,i)   The maximum (right-hand) z-coordinate of the ith special region, ZMA(i), where ZMI$\leq$ZMA$\leq$ZMAX. Default: 0.0

NSUR NSURF NTOFF

Integer variables which give the number of surfaces where all "surface type" histograms will be accumulated, and whether time-of-flight histograms will also be included. See Section 10.3.2 for the list of the histograms and their IDs. The location of each surface is given in the accompanying **RZTS** data card. The surfaces must have cylindrical symmetry, being either circles in the x-y plane (perpendicular to z-axis), or cylindrical shells oriented along the z-axis. Time-of-flight specifications are given in the accompanying **TOFF** data card.

NSURF   The number of surfaces, $0 \leq$NSURF≤ 100. Default: 0

NTOFF   Turns on the generation of time-of-flight distributions for all declared surfaces when NTOFF=1. Default: 0

RZTS (RZTSUR(4,i) UNITi), i=1,100)

Four real values which specify the location and size of each surface, and an integer giving an optional unit number for recording the particles crossing each surface. This card is valid only if NSURF in the **NSUR** card is ≥ 1. See Section 4.6 for an example.

RZTSUR(1,i)   The minimum radius of the ith surface, $0.0 \leq$minimum radius$\leq$RMAX. Default: 0.0

RZTSUR(2,i)   The maximum radius of the ith surface, minimum radius$\leq$maximum radius$\leq$RMAX. Default: 0.0

RZTSUR(3,i)   The minimum z-coordinate of the ith surface, ZMIN$\leq$minimum coordinate$\leq$ZMAX. Default: 0.0

`RZTSUR(4,i)`   The maximum z-coordinate of the ith surface, minimum coordinate$\leq$maximum coordinate $\leq$ZMAX. Default: 0.0

`UNIT(i)`   An optional real, `N`, to open a file named FORT.N which holds a list of all particles crossing the ith surface. `81.`$\leq$`N`$\leq$`90.`. If `N=0.`, then no file will be created. Default: 0.

TOFF `TOFMIN TOFMAX TOFSHF`

Real variables which define the time interval, in seconds, for the time-of-flight spectra accumulated for the surfaces declared by the **NSUR** and **RZTS** data cards, when `NTOFF=1`.

`TOFMIN`   The begining of the time interval. Default: 0.0

`TOFMAX`   The end of the time interval. Default: 1000.0

`TOFSHF`   A time shift applied to the time spectra. Default: 0.0

NHBK `NHBK`

Integer variable which defines the number of materials for the "global" energy deposition histograms accumulated for the materials declared by the **HBKE** data card. There must be at least one declared material, entering `NHBK 1` into MARS.INP, when user-defined histograms are used, via the subroutines MHSETU and MFILL, even if standard histograms are not requested (NOB = NSUR = 0), and even if the user is not interested in the "global" energy deposition for the user defined histograms.

`NHBK`   The number of materials. `1`$\leq$`NHBK`$\leq$`5`. Default: 0.0

HBKE `RIM(i),EHMIN(i),EHMAX(i),i=1,NHBK`

Real variables which defines, correspondingly, the material index, lower and upper energy deposition histogram boundaries for the total energy N(ED) (GeV) deposited in the entire system (RIM=0.) and in materials with index RIM>0 for histograms with $11\leq$ID≤15. Default: 0.0

`RIM`   The material index. `0.`$\leq$`RIM`$\leq$`100.`. Default: 0.0

`EHMIN`   The beginning of the energy deposition interval (GeV). Default: 10^{-6}

`EHMAX`   The end of the energy deposition interval (GeV). Default: $1.5 \times E_0$

NCLD `NCLD`

Integer variable which sets the number of materials on the `IMNC` card in which detailed 2-D mass-charge nuclide distributions are calculated.

`NCLD`   The number of materials on the `IMNC` card for nuclide production calculations. `1`$\leq$`NCLD`$\leq$`5`. Default: 1.

IMNC `IMNC(5)`

Integer variables which specify the material indicies where detailed 2-D mass-charge nuclide distributions are calculated. Example: `IMNC 1 5 14` to calculte and print nuclide production in first, fifth and fourteenth materials with a `NCLD` card as `NCLD 3`.

`IMNC(i)`   The index of the ith material. Default: 1,4*0.

NDET `NDE1`

Integer variable which sets the number of point-like detectors for the spectrum, flux and energy deposition of low-energy neutrons. The size and location of each detector is specified by the accompanying **FLOC** data card. Active only when `IND(5)=F`.

NDE1   The number of detectors. $0 \leq$ `NDE1` ≤ 10. Default: 0

FLOC `RD XD(10) YD(10) ZD(10)`

Real variables which specify the size and location of each of the declared point-like low energy neutron detectors. Active only when `IND(5)=F` and `NDE1` ≥ 1.

RD   The radius of all declared detectors. Default: 0.5.

`XD(i)`, `YD(i)`, `ZD(i)`   The coordinates of the ith detector. Default: 30*0.0

TAPE `NWEGH NWANS NWMJL NWPSI NWNEUN`

Integer flags which control the creation of various output files.

NWEGH   Only one of the following files is generated in any single job, depending on the value. Default: 0
= 17 Write a `MUON.EGH` file with muon-generated EMS for post-run processing and analysis.
= 18 Write `TRACK.PLOT` file with particle tracks.
= 19 Write `VERTEX.PLOT` file with particle interaction vertices.
= 20 Write `MUON.PLOT` file with muon interaction vertices.

NWANS   When set = 2, write `ANSYS.ED` file of energy deposition density to be used by the `ANSYS` system. Default: 0

NWMJL   When set = 12, write `EDMJL.GRA` graphics-oriented file of energy deposition density. Default: 0

NWPSI   When = 13, write `PSINEU.GRA` graphics-oriented file of neutron spectra. When = 14, write `LENEUTRONS` file of neutrons with E$\leq$`RLCTNE(IM)`=0.0145 GeV at their origin in material with index IM for further transport with a standalone code such as `MCNP`. These neutrons are not tracked in `MARS` then, just dumped to the file instead. Default: 0

NWNEUN   When set = 3, write `NEUTRINO` file of generated neutrinos at their origin. Default: 0

4.2.7   Physics Control

ALMX `ALMAXB(5)`

Maximal angles on a step due to magnetic field for five energy intervals.

`ALMAXB(5)`   Real numbers specifying the maximal angles (rad) allowed on a step due to magnetic field for five energy intervals $\Delta = |E_0 - E|/E_0$: <0.001, 0.001-0.01, 0.01-0.3, 0.3-0.999, >0.999. Default: 0.001,0.002, 0.005, 0.01,0.03.

ICEM `C1CEM C2CEM EMODEL IQGSM NEVTYPE`

Variables which control hadron event generator and use of cascade-exciton and quark-gluon string models, and evaporation scheme.

`C1CEM`, `C2CEM`   Real numbers specifying the energy ECEM below which the CEM2003 code is called on the nucleus with atomic number $Z > 4$: $ECEM = C1CEM - C2CEM \times Z$. Default: 5, 0.

EMODEL   Real number specifying transition from high-energy and low-energy event generator algorithms. Default: 5 GeV.

IQGSM   An integer that switches between the inclusive and exclusive event generators at nuclear inelastic interactions. For IQGSM = 0, the default inclusive model is used above the $ECEM$. For IQGSM = 1, exclusive modeling with the LAQGSM2003 code is done at $ECEM < E <$800 GeV for all A, and at $E <$800 GeV and $A \leq 2$ for $\overline{p}, K^{\pm}, d, t,^3 He,^4 He, \overline{n}$, hyperons and heavy ions. For IQGSM = 2, exclusive modeling with the LAQGSM2003 code is always done for everything (currently for photonuclear reactions only at $A \leq 2$ and $0.02 < E < 2$ GeV).

NEVTYPE   The number of evaporated particle types to consider in each nuclear interaction. Minimum is six $(p, n, d, t,^3 He,^4 He)$, maximum is 66 (up to ^{28}Mg). Default: 17 (up to ^{12}Be).

MCSD KMCS KDLE MCSDLE

Control of the ionization energy loss processes and scattering acting on muons, charged hadrons and heavy ions: nuclear elastic scattering (HES) at E>0.014 GeV for neutrons and at all energies for all other particles, and multiple Coulomb scattering (MCS) for all charged particles. This feature is primarily for study purposes, to isolate and study the modeling of specific phenomena. Aside from such studies, users are advised to leave these parameters at their default values, although the use of KDLE=0 can save a fare amount of CPU-time spent tracking high energy muons, if the details of the produced muons are not important to the user's results.

KMCS   An integer whose value turns on or off the elastic and Coulomb scattering. For KMCS = -1, scattering is turned off. For KMCS = 0, only hadron elastic scattering is turned on. For KMCS > 0, both hadron elastic scattering and Coulomb scattering are turned on, and in this case the value k of KMCS gives the number of levels of the particle cascade tree to which the scattering applies. (Default: KMCS=10).

KDLE   Integer number controlling the ionization energy loss modeling. For KDLE=0 a simple continuous dE/dx process is applied. For KDLE>0 a more sophisticated algorithm is applied, which includes delta-electron production and direct e+e- pair production, and the simulation of the resulting induced electromagnetic showers for the first $k \leq$KDLE-levels of the hadron cascade tree and for all muons. (Default: KDLE=15).

MCSDLE   Integer number controlling the correlation in ionization energy and Coulomb scattering. For MCSDLE = 1, correlated energy loss straggling and Coulomb scattering are modeled for all charged particles. For MCSDLE = 0, these two processes are modeled independently. Default: 1.

MUON KPHA IXCL IMUDUM ISOURCE IDNDX

Integer variables which control features of muon and neutrino beams, interactions and tracking.

KPHA   The number of hadron generations to follow, i.e. the number of levels in the hA vertex tree. Default: 30

IXCL   A flag which activates, when IXCL=1, the forced π, K-decays, and $\overline{p}$-production. At IXCL=0 the decays are sampled analogously. Default: 0

IMUDUM   Reserved.

47

ISOURCE   A parameter which defines the beam type: original at `ISOURCE=0` and `ISOURCE=1`, broad at `ISOURCE=2`, μ-decays in a straight section ($z < 0$, at $ZMIN - 0$) at `ISOURCE=3`, and μ-decays in a ring at `ISOURCE=4`. Default: 0

IDNDX   A flag which activates, when `IDNDX=1`, the accumulation of muon angular distributions in all declared `NOB` special regions, when `IND(10)=T`. Default: 0

NUFR `ZLNUFR ZUNUFR RUNUFR`

Real variables which define a cylindrical region where ν-interactions are forced.

ZLNUFR, ZUNUFR, RUNUFR   The cylindrical region where neutrinos are forced to interact with matter when `IND(8)=T`. Default: 0., 0., 0.

PHOT `ELSYN EMISYN EEGHM IPHOTO ISYWRT`

Variables which control features of electromagnetic particle interactions.

ELSYN   Real number specifying the synchrotron emission for electron energy, `E>ELSYN`. Default: 0.5 GeV.

EMISYN   Real number specifying a synchrotron photon energy threshold. For `E>EMISYN` induced EMS are treated at photon energy; otherwise that energy is deposited locally. Default: 0.005 GeV.

EEGHM   Real number specifying an energy threshold. For `E>EEGHM`, full simulation of muon, δ-electron and e-production and their showers along hardon tracks, local deposition otherwise. Default: 0.001 GeV.

IPHOTO   Flag for photo-production. For `IPHOTO=0`, sampled neutron photo-production; forced neutron photo-production for `IPHOTO=1`. Default: 0

ISYWRT   Flag which activates, when `ISYWRT=1`, writing of parameters of first 30000 generated synchrotron photons into a `fort.79` file. Default: 0

RZMN `RMI514 RMA514 ZMI514 ZMA514 RMINTR RMAXTR ZMINTR ZMAXTR`

Real variables which define two cylindrical volumes; each volume is defines by minimum and maximum radii, and minimum and maximum z coordinates. The first four variables define a volume where the `IND(5)=T` condition applies. The second four variables define a volume where the particles entering are recorded in `TRACK.PLOT` and `VERTEX.PLOT` files, if those files have been requested via the **TAPE** data card.

RMI514, RMA514, ZMI514, ZMA514   The cylindrical region where `IND(5)=T` applies. Default: 0.0 RMAX 0.0 ZMAX

RMINTR, RMAXTR, ZMINTR, ZMAXTR   The cylindrical region where particles are recorded. Default: 0.0 RMAX 0.0 ZMAX.

4.2.8   Termination

STOP Terminates the data card list. Any cards given after this are meaningless, unless one runs the MARS-MCNP version and puts the material description required by MCNP after the `STOP` card.

4.3 MCNP Material and Geometry Input

There are two options in the MARS15 code linked to the MCNP libraries. Both are activated by creating and running the *rmars-mcnp-fems-OS* executable rather than default *rmars-bnab-fems-OS*, and by putting INDX 5=T and a MCNP section into the MARS.INP file. At INDX 5=T, this section is put after the STOP card, between two special cards, *MCNP START and *MCNP STOP. For more details see MARS-MCNP bootstraping section.

First option is a substantially enhanced treatment of neutron transport below 0.0145 GeV with accurate description of all the corresponding physics (including photon and recoil proton/fragment production) from the MCNP4 code, with all the transport and geometry handling done by MARS. In this case, only material correspondece cards are included into the MCNP section of the MARS.INP file. Second option is in addition to use a geometry description in the MCNP code format. Activated by INDX 5=T 16=T, the MARS15 code will directly use a MCNP geometry deck for its geometry module and perform a full MARS physics modeling in this geometry, optionally combined with other geometry descriptions in the same setup. Such a MCNP section after the STOP card may look like:

∗MCNP START
10 1 -0.00129 1 -2 -3 imp:n,p=1
20 2 -7.87 1 -2 3 -4 imp:n,p=1
90 0 -1:2:4 imp:n,p=0

1 pz 0
2 pz 100
3 cz 5
4 cz 5.2

mode n,p
m1 7014 -0.7494 8016 -0.2369 18000 -0.0129 1001 -0.0008
m2 26000 1
vol 7854 640.9
∗MCNP END

Each of the cards here describe a single line of free-format input up to 80 characters and each of the cards can have continuation card(s). Comment cards can be used also for convenience. All the MCNP input data used in the MARS.INP file fall into the following categories: (1) materials and other data (volumes and problem type); (2) zones (cells in MCNP terminology); (3) surfaces. The blank card delimiter must be placed between the three groups. If the first option is used in the MCNP mode with native MARS geometry, then only the first group of the cards is required in its truncated form and without any delimiters, i.e.

∗MCNP START
m1 7014 -0.7494 8016 -0.2369 18000 -0.0129 1001 -0.0008
m2 26000 1
∗MCNP END

Otherwise, with MCNP geometry (IND(16)=T), each of the three card groups is mandatory.

The above sample input, representing a segment of a beam-pipe filled with air, describes two cylindrical regions 100-cm long, situated from 0 up 100 cm along z-axis. In radius the regions extend from 0 up to 5 cm and from 5 up to 5.2 cm and are filled with air and iron, respectively. The region outside of the external cylinder is a blackhole (in MCNP terminology - region of zero importance). Volumes of the zones (in cm^3) are entered on the last card of the third group (vol-card). For zones having a cylindrical symmetry MCNP itself calculates volumes in which case the vol-card is not required. For uniformity, it can be omitted when calculating all region volumes in the setup in a short Monte-Carlo session (IVOL = 1). The card $modenp$ is left just for convenience reasons.

Comprehensive description of the formal rules to construct MCNP geometry and material input can be found in MCNP manual [.]. Briefly the rules are given in two following subsections.

4.3.1   MCNP Material Data

The cards with material data are numbered with positive integers in arbitrary order and with any gaps. MARS uses only the sequential numbers of the cards which must correspond to the sequence of the same materials according to entries on MATR card in MARS.INP.

Isotopes in MCNP are described in terms of ZZAAA notation. First two digits give the atomic number while the last three ones give the atomic mass. Thus, 92238 describes uranium-238, 5010 describes boron-10, etc. If each of the last three digits equals to zero, then the natural mixture of isotopes is assumed. Materials in MCNP are represented as mixtures. For example, the line

m1 26000 0.8 6012 0.2

describes the first material (in the MARS numbering scheme!) with 80% of atoms being natural iron (26000) and 20% of atoms being carbon-12 isotope. Second example is for the same material, but the percentage in the mixture is given by weight:

m1 26000 -0.95 6012 -0.05

Vacuum regions are described in MARS either with a material index $IM = 0$, or with a 'VAC' entry in a corresponding section of the MARS.INP file. If the problem uses such a 'VAC' material, the following convention applies to the corresponding material card in MCNP input section of MARS.INP: the card should describe an arbitrary material that complies with the MCNP format. At a preprocessing stage this particular material will be substituted with vacuum.

The atom or weight fractions entered within a material card can be unnormalized. MCNP performs re-normalization by default. All entries within such a card must be either weight or atom fractions, i.e. combinations of atom fractions with weight ones within a card are not allowed.

The file xsdir can contain several references to different evaluated nuclear data sets for the same material. The data could have been prepared by different scientific groups with different accuracies and at different conditions like temperature of the material etc. By default, at initialization stage for each material (e.g. 26000) MCNP reads in the first reference to corresponding nuclear data set encountered in the file xsdir. To differentiate between the data sets, specific suffixes are used. For example, if one

uses a data card

    m1 26000.42c 1

exactly the specified nuclear data set will be used for iron (see MCNP manual [.] for details).

4.3.2   MCNP Geometry Input

The MCNP geometry can be thought of as another type of Extended Geometry. At the same time it is an independent powerful tool for geometry description. MARS15 supports all of the MCNP geometry capabilities except lattices and universes.

In a MARS geometry model there must always be at least one Standard zone, and all the declared MCNP zones must be contained within the outermost Standard zone boundary. MCNP zones are numbered starting from $NFZP + 1$ where $NFZP$ is the number of Standard zones. There is no explicit restriction on maximum number of MCNP zones and finally the number is determined by parameter $MDAS$ at the stage of MCNP-to-MARS bootstrapping [.].

First group of the cards in the sample above (cell cards in MCNP terminology) describe each zone as a spatial region limited with a set of surfaces. Each surface, described with an equation

$$f(x, y, z) = 0,$$

divides entire space into two regions, so-called positive and negative spaces that include all points satisfying inequalities $f(x, y, z) > 0$ and $f(x, y, z) < 0$, respectively. Finally each zone is constructed using the surface numbers with signs (sign $+$ is optional while $-$ is mandatory) to indicate which of the spaces, positive or negative, is assigned to the zone. In addition, boolean operators AND and/or OR are used. One or more blanks between any two surface numbers means boolean operation AND applied to their spaces to form an intersection of the spaces, while colon means boolean operation OR to form a union of the spaces. Intersections are performed first and then unions. Complicated structures with changed hierarchy of the boolean operations are allowed. Arbitrary intersections and/or unions of the spaces are allowed with the only restriction: entire space of a zone should have definite sign relative to each of its limiting surfaces, or in other words, the space can not be partly positive and partly negative relative to any of its limiting surfaces.

Each zone card contains also material number (see subsection ..) and numerical value of the density of the material which the zone is filled with. Negative entry means mass density in units of g/cm^3, while positive entry is used for atom density in units of 10^{24} atom/cm^3. A zone card contains also relative importance of particles in the zone. By default the values of importances are set equal to unity. For a card describing a blackhole both material number and importance must be zero while material density must be omitted.

Surfaces used to construct spatial zones are listed in the second group. Each of the cards contains mnemonic name of the surface and numerical entries (in cm and dimensionless depending on the surface). The second card of the second group represents a plane orthogonal to z-axis and described with the equation $z - 100 = 0$. The fourth card of the group describes an infinite cylinder of 5.2-cm radius

51

and with its axis coinciding with z-axis. The total set of regular (in the sense of analytical geometry) surfaces allowed is the following: plane, sphere, cylinder, cone, ellipsoid, hyperboloid, paraboloid, and elliptical or circular torus. In addition, it is possible also to use so-called macrobodies, treated as regular surfaces, which are composed of the regular surfaces mentioned, just like a box is composed of six different planes. The following macrobodies are allowed at present: arbitrarily oriented orthogonal box, rectangular parallelepiped, sphere, right circular cylinder, right hexagonal prism. For convenience, a surface can be defined in a local coordinate system other than the global coordinate system of the problem under consideration. In such a case a transformation matrix between the two systems must be supplied by the user.

Both the zone and surface numbers can be positive integers ordered in any way and with any gaps. MARS assigns its own non-standard zone numbers according to the sequence of the zones in the first group of the cards.

4.4 Extended Geometry Input

Extended Geometry zones are an alternative way to describe the user's model. The term "extended" refers to an extension beyond the r-z-ϕ symmetric Standard zones which are described by the ZSEC and RSEC cards in the MARS.INP file. Extended Geometry zones are constructed from a set of contiguous or overlapping geometrical shapes, currently, boxes, cylinders, spheres, cones, and tetrahedra. This is similar to the methods used by GEANT particularly. There can be up to 10^5 extended volumes up to 500 arbitrary transformation matricies, and volumes can be sub-divided into sub-regions. In this mode, small regions within a large volume are prevented from being skipped over.

The details of the arrangements of the geometric shapes are controlled by lines in the input file GEOM.INP. As indicated in Section 4.2.1, setting IND(3)=T in the MARS.INP file activates the reading of the GEOM.INP file and implementation of the Extended Zones described by it.

The user is reminded that the overall outer boundary of the model ("mother volume") is set by the ZSEC and RSEC cards in the MARS.INP file. There must always be at least one Standard zone, and all the declared Extended zones must be contained within the outermost Standard zone boundary. In assigning zone numbers, MARS begins with the Standard zones, numbered 1 : NFZP, the maximum number of declared Standard zones. Extended zones are numbered starting from NFZP+1. There can be no more than $100,000$ declared Extended zones. See the discussion in Section 3.1.

The code reads volume data defined in their own local coordinate systems (LCS) in the input file GEOM.INP.

First line of the GEOM.INP file is a title, consisting of no more than 80 characters. All the subsequent lines are data, either *volume* lines or *transform* lines. A *volume* line is used to declare and describe a single Extended Geometry Volume. A *transform* line describes a transformation matrix, translation and rotation, which can be applied to any Extended geometric volume. Each volume can be subdivided into a number of subregions. Blank lines can be inserted between data lines, for file organizational purposes, and will be skipped by the routines reading the file. Comments are allowed at the line end starting from an exclamation sign. The Extended volume description lines can be listed in any order, for example they are not required to be listed from the inner-most to the outer-most; however the MARS zone number will be assigned according to the order in which the lines appear in the file.

Each Extended volume is specified by a single line of data; the data is unformatted, and separated by blank spaces. The syntax of an Extended volume description line is

```
VNAM NT NTR IM XR YR ZR C1 C2 ...,
```

where

`VNAM` is volume name (A8);

`NT` – volume type; $1\leq$`NT`≤ 5;

`NTR` – transformation matrix number; 1 stands for the transformation matrix described on the `TR1` card, 2 stands for the transformation matrix described on the `TR2` card, etc; $0\leq$`NTR`≤ 500;

`IM` – material number (index) of given volume; $0\leq$`IM`≤ 100;

`XR`, `YR`, `ZR` – coordinates of a reference point (RP) of the LCS for given volume;

`C1`, `C2`, `...` – parameters of given volume, including the number of subdivisions (see Table 4).

Transformation matricies are implemented via `TR`-cards whose syntax is

```
TR1 XT1 YT1 ZT1 α1 β1 γ1
```

```
TR2 XT2 YT2 ZT2 α2 β2 γ2
```

where

`TR1`, `TR2`, `...` are transformation matrix IDs;

`XT1`, `YT1`, `ZT1`, `XT2`, `YT2`, `ZT2`,`...` are translation coordinates;

$\alpha 1$, $\beta 1$, $\gamma 1$, $\alpha 2$, $\beta 2$, $\gamma 2$,`...` are rotation Euler angles (in degrees) around x, y and z axes clockwise (positive) or counterclockwise (negative). If there is a rotation around more than one axis, the rotation sequence is γ (around z-axis), then β (around y-axis), and then α (around x-axis). Total number `NTRAF` of transformation matricies is $0\leq$`NTRAF`≤ 500.

The mother volume defined in `MARS.INP` contains all other volumes in $standard$, $extended$, $MCNP$ and user-supplied (`REG1`) geometry sectors. Usually, this is a "big" cylinder (most often with `IM=0`). A current set of the extended volume types and description parameters are given in Table 4.

A `STOP`-card whose syntax is

```
STOP
```

terminates the GEOM.INP data card list.

Table 4 : Current set of extended volume types in GEOM.INP and their description in LCS. Arbitrary translation and rotation in the GCS are allowed via corresponding transformation matricies defined by the TR cards.

NT	Name	Description
1	BOX	C1 and C2– half-sizes along x and y axes,
		C3– box length along z axis,
		C4=NXSB – number of subdivisions along x-axis, default: 1;
		C5=NYSB – number of subdivisions along y-axis, default: 1;
		C6=NZSB – number of subdivisions along z-axis, default: 1;
		RP– center of the lowest z plane;
		x and y LCS axes are parallel to two perpendicular
		edges of the lowest z plane;
		z axis points from center of the lowest z plane to
		center of the highest z plane;
		x, y, z axes of the LCS are parallel to those of the GCS.
2	CYLINDER	C1 and C2– inner and outer radii of the cylinder;
		C3– cylinder length along z axis,
		C4=NZSB – number of subdivisions along z-axis, default: 1;
		C5=NRSB – number of radial subdivisions, default: 1;
		C6=0;
		RP– center of the lowest z plane;
		z axis points from center of the lowest z plane to
		center of the highest z plane;
		z axis of the LCS is parallel to that of the GCS;
		x and y axes of the LCS orientation is arbitrarily.
3	SPHERE	C1 and C2– inner and outer radii of the sphere,
		C3=NRSB – number of radial subdivisions, default: 1;
		C4=C5=C6=0;
		RP– sphere center;
		LCS axes oriented arbitrarily.
4	CONE	C1 and C2– inner and outer radii at the lowest z;
		C3 and C4– inner and outer radii at the highest z;
		C5– cone length along z axis,
		C6=NZSB – number of subdivisions along z-axis; default: 1;
		C7=NRSB – number of subdivisions in a radial direction; default: 1;
		RP– center of the lowest z plane;
		z axis points from center of the lowest z plane to center
		of the highest z plane;
		z axis of the LCS is parallel to that of the GCS;
		x and y axes of the LCS orientation is arbitrarily.
5	TETRAHEDRON	RP– $x1, y1, z1$ coordinates of the first reference vertex
		with an optional rotation around it;
		C1 through C9 – coordinates of three other verticies
		$(x2, y2, z2)$, $(x3, y3, z3)$,$(x4, y4, z4)$.

By default, there is no thorough optimization in Extended Geometry, i.e. the internal flag OPT is off. In this case, the elements (shapes) are analyzed in the order as they appear in the GEOM.INP file. That is the sooner a shape appears the higher priority it has. It means, if a point is in element No. 1, all consequent elements do not matter, i.e. a higher priority shape overrides all others. In such a case of overlapping elements, a user has to recalculate the volumes of shapes, using the $IVOL = 1$ and the RZVL card.

There is an optimization in the Extended Geometry mode while there is no overlapping elements. It is turned on by supplying the card OPT in the GEOM.INP file. If the option is provided, the first element checked is the old one where previous hit occurred. It is up to user to check that no shapes are overlapped.

Here are examples of overlapping/overwriting:

```
Order in GEOM.INP      Result

box1
box2

                    +-----------+
                    |           +--------+
                    |    2      |   1    |
                    |           |        |
                    |           +--------+
                    |           |
                    +-----------+

box2
box1

                    +-----------+
                    |           |-----+
                    |    2      |  1  |
                    |           |     |
                    |           |-----+
                    |           |
                    +-----------+
```
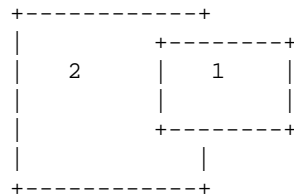
4.5 XYZ Histograming Input

Flexible histograming – called XYZ – is implemented in boxes arbitrary positioned in a 3D-system modeled. This histograming is totally independent of geometry description, both type (standard, non-standard, extended and MCNP-type) and details, including materials distribution. By default, the macro box (detector) axes are parallel to those of the mother volume, and can be rotated using MARS2BML and BML2MARS routines.

The histograms can overlap, contrary to the volume (RZ) and surface histograming. Arbitrary voxels (the smallest distinguishable box-shaped part of a three-dimensional space) can be defined by individual binning NX, NY, NZ in each macro box. Arbitrary cut-off energies can be defined for each flux-type histogram. Histogram ID reserved: $701 < ID < 1000$. These histograms are normalized to $AINT(p/s)$, contrary to all previous histograms and most values in MARS.OUT and MTUPLE which are normalized to one incident particle. Up to 23 different histogram types can be defined in a run as described in Table 4.

Table 4: Histogram types in XYZ-histograming.

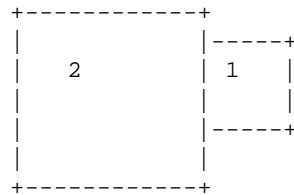| Codename	Value	Units
ST	star density (E>50 MeV)	cm^{-3}s^{-1}
DR	30d/1d residual dose on contact	mSv/hr
FT	total flux of hadrons E>ETFT	cm^{-2}s^{-1}
FH	flux of charged hadrons E>ETFH	cm^{-2}s^{-1}
FN	flux of neutrons E>ETFN	cm^{-2}s^{-1}
FM	flux of muons E>ETFM	cm^{-2}s^{-1}
FG	flux of photons E>ETFG	cm^{-2}s^{-1}
FE	flux of e- and e+ E>ETFE	cm^{-2}s^{-1}
DE	FTD prompt dose equivalent	mSv/hr
DA	absorbed dose	Gy/yr at 2×10^7 s/yr
DP	DPA	DPA/yr at 2×10^7 s/yr
PD	power density, total	mW/g or Gy/s
PH	power density, charged hadrons	mW/g or Gy/s
PN	power density, neutrons	mW/g or Gy/s
PM	power density, muons	mW/g or Gy/s
PG	power density, photons	mW/g or Gy/s
PE	power density, e- and e+	mW/g or Gy/s
SPN	neutron energy spectrum	GeV^{-1} cm^{-2} s^{-1}
SPP	proton energy spectrum	GeV^{-1} cm^{-2} s^{-1}
SPK	pion/kaon energy spectrum	GeV^{-1} cm^{-2} s^{-1}
SPM	muon energy spectrum	GeV^{-1} cm^{-2} s^{-1}
SPG	photon energy spectrum	GeV^{-1} cm^{-2} s^{-1}
SPE	e+e- energy spectrum	GeV^{-1} cm^{-2} s^{-1}

In any run, the total number of histograms can not exceed 300, i.e. a sum over all histogram types in all macro boxes (detectors) must be less than 300. Maximum number of voxels in any macro box (bins in each histogram) can not exceed 10^5. To save CPU-time, keep the number of macro boxes (detectors), histogram types and voxels (bins) as small as possible!

Histograms are defined in an input file XYZHIS.INP, where the following conventions are used:

1. The first line is a textual problem description.

2. The last line, STOP or stop, terminates the XYZHIS.INP data card list, i.e. everything after this line is ignored.

3. Blank lines can be inserted between data cards, for file organizational purposes.

4. Comments are allowed at the line end starting from an exclamation sign.

5. The data is unformatted and separated either by blank space or by one of the following delimiters ", ; : &".

6. A macro box (detector) is defined by a XYZ-card of the following format:

   $XYZ X1 X2 Y1 Y2 Z1 Z2 NX NY NZ TEXT,$

   where XYZ or xyz is a card ID;
   $X1, X2, Y1, Y2, Z1, Z2$ are the minimum and maximum Cartesian coordinates defining the box;
   NX, NY, NZ are the number of bins along each axis;
   $TEXT$ is a word describing the box (optional).

7. Macro box (detector) categories are

   $nxyztp = 1- > X - Y$ in a Δz slice $(x = v, y = h)$, defined by $NZ = 1$
   $nxyztp = 2- > X - Z$ in a Δy slice $(x = v, z = h)$, defined by $NY = 1$
   $nxyztp = 3- > Y - Z$ in a Δx slice $(y = v, z = h)$, defined by $NX = 1$
   $nxyztp = 4- > $ Energy spectra in the macro box, defined by $NX = NY = NZ = 1$

8. A card following a XYZ-card contains a list of histogram types for this macro box: any combination of the above 23 types. An arbitrary cut-off energy can be defined for each flux-type histogram using the following syntax $FN > 0.03$, which means that neutron flux in this particular histogram is scored above 0.03 GeV, not above the default threshold or that defined in the MARS.INP file.

A list of histograms to be filled in the run is generated at the completion of the initialization stage and shows up in an XYZHIS.TAB file which contains all attributes for each histogram. Here is the file created with the above XYZHIS.INP.

Here is an example of a `XYZHIS.INP` file:

```
 XYZ histo test 30-July-2004

xyz -20. 20. -0.4 0.4 0. 50. 40 1 50 XZ-scan_at_|y|<0.4
ST DR FT>0.02 DE DA PD

xyz -0.4 0.4  -10. 10.  0. 50. 1 20 50 YZ-scan_at_|x|<0.4_fluxes
ST FT>0.02 DE DA FH FN FE FG DP

xyz -5. 5. -2. 8. 12. 13. 10 5 1 XY-scan_in_a_z=12-13_cm_slice
FT DP

xyz -5. 5. -5. 5. 12. 13. 10 5 1 ! Muon flux profile at shower max
FM

xyz 0. 5. 2. 7. 10. 15. 1 1 1 Spectrum-1_in_a_5x5x5_cm_cube
SPP SPN SPK SPM SPG SPE

!xyz -5. 5. -5. 5.. 10. 20. 1 1 1 ! Muon-spectrum
!SPM

stop
```

4.6   Input Deck Examples

The following examples use only the input deck to fully describe the problem to be modeled. The user subroutines are not utilized and left as dummies. In truth, most problems of interest will involve the user subroutines; however, these examples serve to outline the basics of the input deck. More complex examples which utilize the user subroutines in conjunction with parameters in the input deck are given in Section 5.17.

4.6.1   Z-Sandwich Geometry

Z-sandwich geometry is one where the same material extends outward for all values of R. The material changes only along Z boundaries. R-sandwich geometry is one where the same material extends along the Z-axis, and the material changes only at concentric R boundaries. The user must choose either one or the other to represent the Standard zones in his model.

Figures 12 and 13 show a simple Z-sandwich Standard geometry. The pictures are produced by the Mars GUI interface. Figure 12 is a color-filled diagram of the Standard zones, with each color being a distinct material: light blue is air, orange is steel, and tan is concrete. Figure 13 is the same zone diagram, with no color fill, and with a 2-D histogram of the charged hadronic flux density superimposed over it.

The input deck for this example is :

```
MARS15 Example:  Z-sandwich geometry
/home/mokhov/restricted/mars15/dat
INDX 2=T 6=T
NEVT 1000
ENRG 100.  0.05 0.05
IPIB 3 3
BEAM 5.0 5.0 0.1 0.1
SMIN 0.2 5.
NMAT 3
MATR 'FE' 'AIR' 'CONC'
NLNG 3
ZSEC 50.  150.  350.  1252=4 4 2501=2 1 3
NLTR 1
RSEC 100.  51=4
NOBL 1
RZOB 0.  100.  0.  350.
STOP
```

At the top of the deck `INDX 2=T` declares this to be a Z-sandwich type geometry, and this means the zone materials are declared in the `ZSEC` card. The `NLNG` and `ZSEC` cards declare 3 major Z sections, with the 2nd and 3rd sections each sub-divided into 4 sub-sections (`1252= 4 4`). The 1st major section is composed of air, the 2nd steel, and the 3rd concrete (`2501=2 1 3`), where the `2 1 3` refer to the order those materials are listed in the `MATR` card. The `NLTR` and `RSEC` cards declare only 1 major R section, which is in turn divided into 4 sub-sections (`51=4`). These sections and sub-sections are delineated by the grid lines seen in Figures 12 and 13. The sections are always defined from minimum to maximum Z, or, from left to right in the Mars coordinate system. The air section is 50cm deep, the steel 100cm deep, and the concrete 200cm deep; the `50. 150. 350.` values listed reflect the locations of the right-hand boundary of each major section.
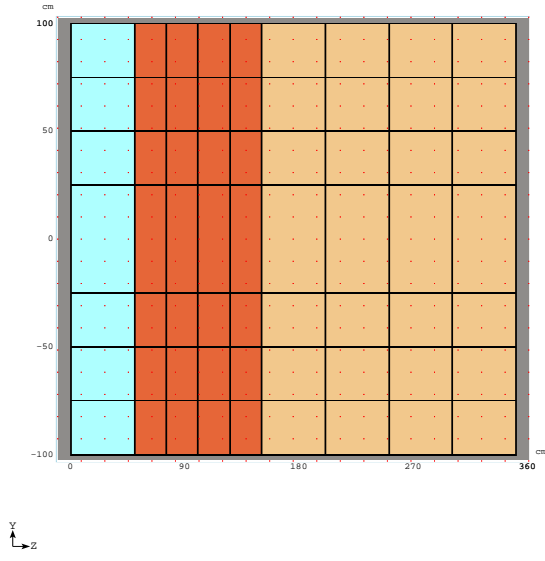
Figure 12: Plan View of the Z-sandwich Example. The grid lines outline the declared Zones.
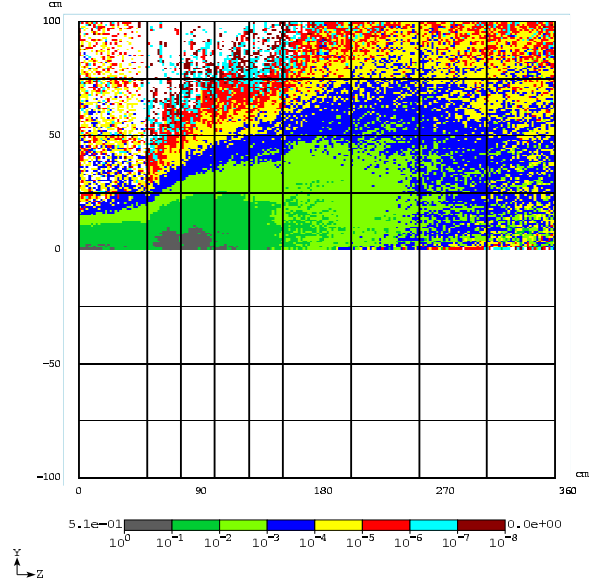
Figure 13: The same Z-sandwich geometry with a histogram of the hadronic flux density overlaid.

The input beam is composed of 100 GeV π^+, and the IPIB and BEAM cards define a fairly large, spreading beam - the example could be a model of a secondary pion beam striking a beam dump. Energy cutoffs of 0.05 GeV are given for both the EM and EPSTAM parameters in the ENRG card. This setting, in conjunction with IND 6=T means that the user is interested primarily in star density and in overall flux density patterns, and not in a detailed analysis of energy deposition. The zone sizes are relatively large, and the global step sizes in the SMIN card are set correspondingly. No special energy cutoffs or step sizes are specified beyond the global values.

The NOBL and RZOB cards declare one volume, encompassing all declared zones. A standard set of histograms then accumulates various parameters, and are written to a mars.hbook file. This file can be opened from the GUI interface, (a separate job from the one which produced the hbook file), and selected histograms projected onto the geometry, as in Figure 13. NOBL histograms are accumulated in a cylindrical geometry, and so display only from R=0 to R=Rmax. The GUI interface displays the geometry in X-Z, Y-Z, or X-Y planes. Thus, the color maps display only in the upper half of the plan or elevation view. The particular histogram shown is a map of the charged hadronic flux, each color being an order of magnitude in flux density. One can see the broad pion beam traversing the air (green and blue color bands), showering and spreading out in the steel and concrete of the beam dump, and some amount of back-scattered particles from the front face of the steel back into the air above the incoming beam. More details on both histograms and the GUI interface are given in Sections 10.3 and 11 respectively.

4.6.2 R-Sandwich Geometry

Next, examine a similar simple geometry, also a beam dump type, but using R-sandwich geometry. R-sandwich geometry is one where the same material extends along the Z-axis, and the material changes only at concentric R boundaries. Again, the user must choose either Z or R sandwich geometries to represent the Standard zones in his model.

Figures 14 and 15 show a simple R-sandwich Standard geometry. The pictures are again produced by the Mars GUI interface. Figure 14 is a color-filled diagram of the Standard zones, with the colors similar to the Z-sandwich example: light blue is air, orange is steel, and tan is concrete. Figure 15 gives a cross-section view of the zone diagram.
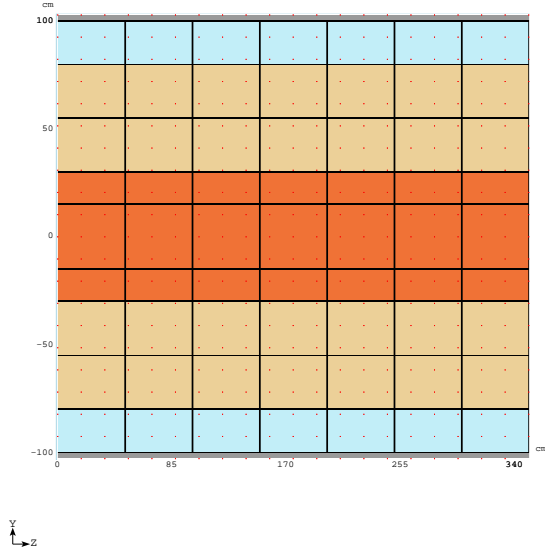
Figure 14: Plan View of the R-sandwich Example. The grid lines outline the declared Zones.
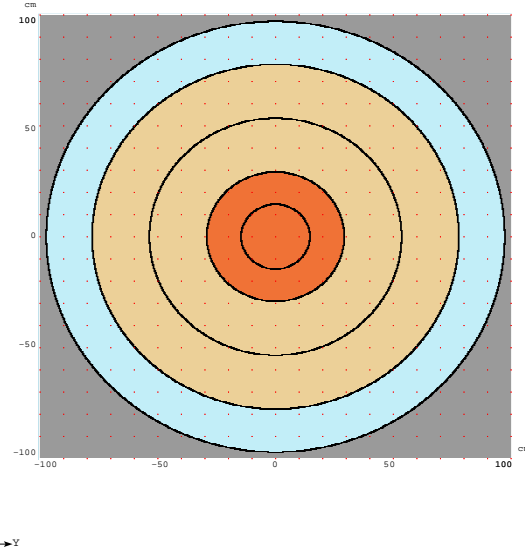
Figure 15: A cross-section view of the same R-sandwich geometry.

The input deck for this example is :

```
MARS15 Example:  R-sandwich geometry
/home/mokhov/restricted/mars15/dat
INDX 6=T
NEVT 1000
ENRG 100.  0.05 0.05
IPIB 3 3
BEAM 5.0 5.0 0.1 0.1
SMIN 0.2 5.
NMAT 3
MATR 'FE' 'AIR' 'CONC'
NLNG 1
ZSEC 350.  1251=7
NLTR 3
RSEC 30.  80.  100.  51=2 2 101=1 3 2
NOBL 1
RZOB 0.  100.  0.  350.
STOP
```

At the top of the deck INDX 2=F declares this to be a R-sandwich type geometry, and this means the zone materials are declared in the RSEC card. The NLTR and RSEC cards declare 3 major R sections, with the 1st and 2nd sections each sub-divided into 2 sub-sections (51=2 2). The 1st major section is composed of steel, the 2nd concrete, and the 3rd air (101=1 3 2). The materials are listed in the same order as for example 1, where the 2 1 3 refer to the order those materials are listed in the MATR card. The NLNG and ZSEC cards declare only 1 major Z section, which is in turn divided into 7 sub-sections (1251=7). These sections and sub-sections are delineated by the grid lines

seen in Figures 14 and 15. The sections are always defined from minimum to maximum R, or, from the center R=0 to the outermost R value. The inner steel section is 30cm in radius, the surrounding concrete section is 50cm thick, and there is a 20cm thick layer or air surrounding the concrete; the `30. 80. 100.` values listed reflect the locations of the outer boundary of each major R section.

The input beam is described identically to the Z-sandwich example. This example could also be a model of a secondary pion beam striking a beam dump, only the emphasis is on mapping the radial deposition outward through the different layers of materials. The same energy cutoffs and global step sizes are used as for the Z-sandwich example. Also the identical `NOBL` and `RZOB` cards are used, encompassing the entire declared geometry. However, since the materials are arranged in a different geometry, the hadronic flux map, Figure 16 appears slightly different from that for the Z-sandwich example. Since the color maps display only from `R=0` to `R=Rmax`, the frame of the view has been modified to show just the upper half of the geometry.
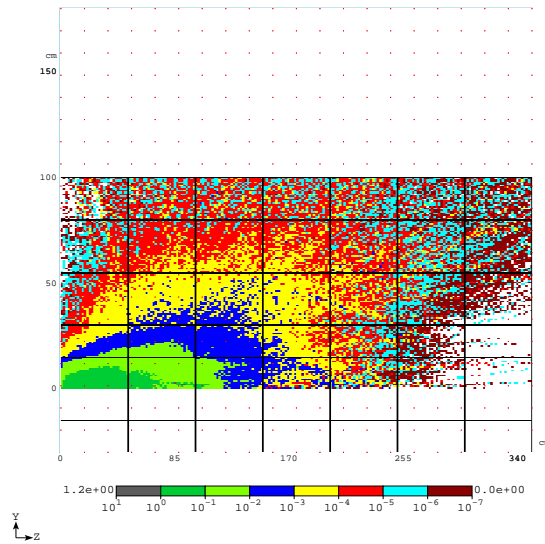
Figure 16: Plan View of the R-sandwich Example, with a map of the hadronic flux overlaid.

4.6.3 Thin Window

The example of a thin window shows the usage of material-dependent parameter settings, such as step lengths. The model is of a small proton beam, traveling through helium. The beam encounters a thin titanium window, and then a region of air. There are no interactions of significance in the helium or the air, and allowing large step sizes in those areas can reduce the overall running time. However, one must be careful of the combination of small zones and large step sizes. It is fairly obvious that if the step size is of the same order as a thin zone size, then some generated particles may "jump over" the thin zone, and some results for the the thin zone, such as energy deposition, will be incorrect. What is less obvious is controlling the accuracy with which the zone boundary is located.

MARS uses a "zig-zag" approach to locate a boundary. The pilot step, the global `STEPH`, or the material dependent `RLSTEH` is first applied; if within that step the zone number changes, then MARS will iterate back and forth, in smaller and smaller steps, to locate the boundary between the zones.

62

The minimum size of these iterative steps is given by the global STEPEM or the material dependent RLSTEM. Therefore, the zone boundary will be located to the accuracy of the size of STEPEM or RLSTEM. Moreover, one does not want to have a large value of STEPEM on one side of a boundary, and a smaller value of RLSTEM on the other side of a boundary. A better strategy is to make big steps until within a certain distance from the thin zone, and then reduce the step size on approach, step through the thin zone, continue taking small steps for a certain distance past it, and then increase the step size again. This requires dividing the material in front of and behind the thin material into sections where the step size can be adjusted.

The following deck sets up this sort of controlled boundary localization:

```
MARS15 Example:  Thin Window
/home/mokhov/restricted/mars15/dat
INDX 2=T
NEVT 1000
ENRG 50.
IPIB 4 3
BEAM 1.0 1.0 0.02 0.02
SMIN 0.2 1.0
NMAT 5
MATR 'HE' 'HE' 'TI' 'AIR' 'AIR'
NLNG 5
ZSEC 20.  22.  22.5 24.5 44.5 2501=1 2 3 4 5
NLTR 3
RSEC 1.  5.  10.  51=4 4 2
MTSM 2=0.01 0.01 0.01
MTSH 2=0.1 0.1 0.1
STOP
```

The deck is set up for Z-sandwich geometry, given by INDX 2=T and by the fact that the assignment of materials to zones is given in the ZSEC card. There are only three unique materials among the five listed, and this is done so that different step sizes can be assigned to the zones of helium and air adjacent to the titanium window. The total amount of helium and air on either side is 22cm, and is split into large zones 20cm deep and smaller zones 2cm deep adjacent to the window. The window itself is 0.5cm thick (not a very thin window, but the example is for illustrative purposes). Each of these five zones is given a unique material ID number. Next, the global step sizes, given by the SMIN card, are sized for the 20cm deep zones of helium and air. The MTSM and MTSH cards adjust the step sizes downward for materials 2, 3, 4, with the step sizes in those 3 materials being the same values in this example. These smaller step sizes are appropriate for the thickness of the thin window; taking similar small steps through the helium and air on either side of the window assures that the window boundary will be accurately located, and data like energy deposition correctly tabulated.

In keeping with a desire for accurate energy deposition data, the deck allows all the default energy thresholds, specifying only the energy of the incoming pion beam; the beam is described as a simple diverging gaussian. Radial zones are defined, in the RSEC card, so that the data can used to show how the energy deposition changes versus radial distance; an example of extracting and plotting this data is given in the output examples in Section 10.1.

Figures 17 and 18 show the plan and cross-section views of the Thin Window described by the input deck. The helium zones are a light purple, and the air zones a light blue, with the lighter tones being the 2cm deep zones just in front of and behind the thin window, where the step lengths are reduced. The thin window itself is a gold color. The cross section view shows the radial zones in the thin window,
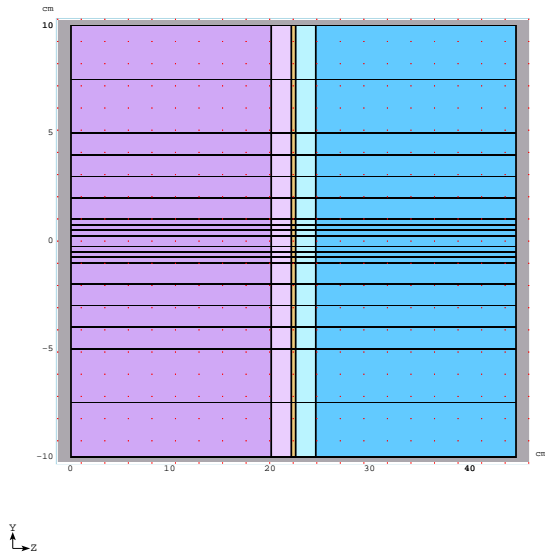
Figure 17: Plan View of the Thin Window example.
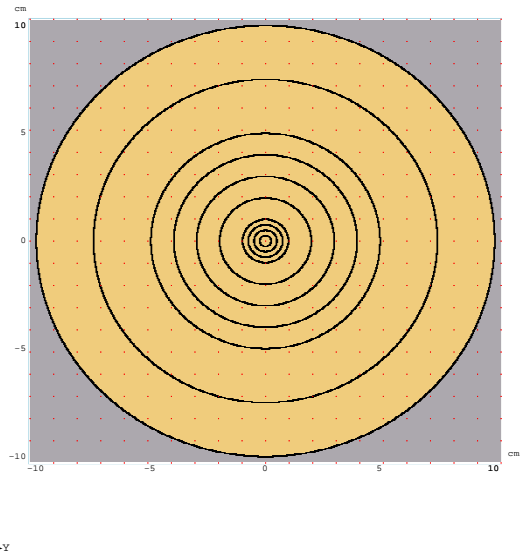
Figure 18: A cross-section view of the Thin Window example.

where data on parameters like energy deposition are accumulated.

4.6.4 Example of Extended Geometry

Option 3. Here is an example of *extended* geometry description. The following is the MARS.INP file:

```
Extended geometry test 10/22/02
/home/mokhov/restricted/mars15/dat
INDX 3=T
CTRL 0
NEVT 5
ENRG 8.  5=0.001
IPIB 1
SMIN 0.01 3.
ZSEC 160.
RSEC 80.  101=0
NMAT 6
MATR 'CU' 'FE' 'AIR' 'CONC' 'CH2' 'CH'
STOP
```

A GEOM.INP file looks as:

```
Exgeo test

ball-0      3   0   0   0.      0.      10.   0.        2.
ball-1      3   0   1   0.      0.      10.   2.        5.
ball2a      3   0   2   0.      30.     15.   0.        4.    2
ball2b      3   0   1   0.      30.     15.   4.        10.   3     !   ball-2b
ball-3      3   0   4   0.      0.      60.   0.        15.   5

box-1       1   0   3   1.      -30.    15.   4.        4.    17.   2
box-2       1   0   5   -3.     -40.    45.   7.        12.   10.   5                    !   box-
cone-1      4   0   6   0.      0.      90.   2.        5.    6.    12.          20.
cone-2      4   1   3   0.      30.     90.   6.        12.   2.    5.           30.
cyl0        2   1   1   0.      30.     90.   0.        2.    30.   2            2
cone-4      4   1   4   0.      30.     90.   2.        6.    2.    2.           30.
cone-3      4   0   0   0.      0.      90.   0.        2.    0.    6.           20.

cyl-1       2   0   1   0.      0.      120.  0.        5.    40.   10           10
cyl-2a      2   0   2   0.      50.     120.  1.        5.    30.   10           2
cyl-2b      2   0   1   0.      50.     120.  0.        1.    30.   !   cyl-2b
cyl-3       2   0   3   0.      30.     130.  0.        5.    20.   4            5
cyl-4a      2   2   4   0.      -40.    125.  0.        5.    25.   5            2
cyl-4b      2   3   4   0.      0.      0.    0.        5.    25.   5            2

TR1         0.  0.  0.  -14.32  0.      0.    !   TRF-1
TR2         0.  0.  0.  -11.46  0.      0.    !   TRF-2
TR3         0.  -60. 115. 11.46 0.      0.    !   TRF-3
stop
```

    Corresponding view is shown in Fig. 19. User subroutines REG1 and REG2 are not needed in
this case. Detailed histogramming can be done via HBOOK (as in *Option 2*) or by adding a required
number of regions into the files MARS.INP or GEOM.INP. Depending on the application an appropriate
combination of the *standard* and *extended* volumes, of the *standard* and HBOOK histogrammings
etc., can improve the outcome.
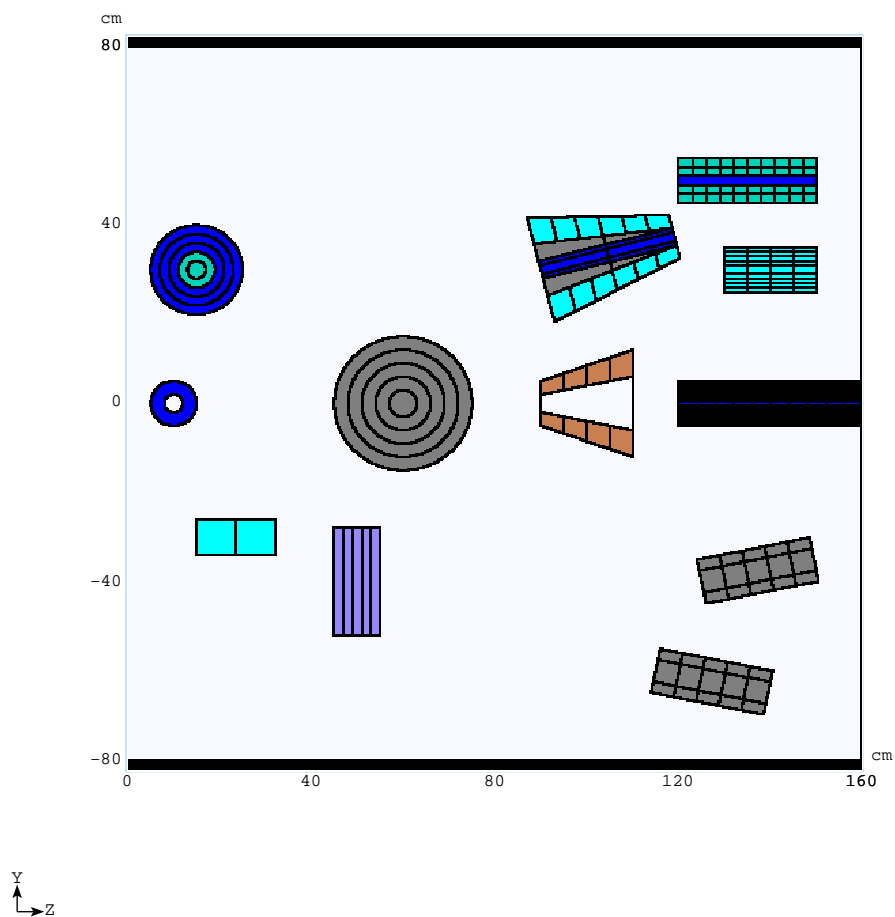
Figure 19: Plan View of the Extended geometry Example.

5 User Subroutines

The user subroutines are collected in the file **m1504.f**. The default version of this file, distributed as a part of the MARS code, and installed in a directory just above the MARS libraries, contains dummy versions of each of the user subroutines. Some of the subroutines contain comments and commented-out sample code as reminders for their use, but the user should consult this manual for details on the full usage of the subroutines.

Table 5 lists the names of the user subroutines, and a brief description of their purpose.

The user is free to split file **m1504.f** up into other files, for example, combining all his customized subroutines into a single file, and storing all the unused dummy user subroutines in another file. The user can also create his own subroutines which are called from the user subroutines. The user can create his own set of common blocks to carry information among his own subroutines; none of his values or variables will be carried back into the main MARS code, however. With a few exceptions, the communication between MARS and the user subroutines is via subroutine arguments.

If the user rearranges subroutines into files other than **m1504.f**, then he must remember to also modify the default GNUmakefile appropriately, so the compiler-linker knows to pick up the local user files. This manual assumes the user knows how to compile and link code using gmake or make on unix or linux systems.

It is important to remember that MARS uses double precision in all calculations, and all real variables are declared double precision. The user will utilize MARS real variables delivered via subroutine arguments, and rather than copy them to local single precision variables, it is highly recommended that all user real variables be likewise declared double precision. This means that all real value assignments must be of the form `XVar=0.24D0` or `Parameter(YLimit = 435.76D0)`. The exception to this is when user-defined histograms are employed; real variables loaded into HBOOK must be single-precision.

Table 5: User subroutines.

Subroutine name	Description
MARS1504	main "initialize-run-write output" steering routine
MIXTUR	define a compound material
BEG1	define the primary particle distribution
REG1	define user-encoded non-Standard zones
REG3	redefine the material content of Standard zones
FIELD	define magnetic fields
SUFI	read in magnetic field data
LEAK	tag particles which reach selected zones
ALIGN	discrete fictitious scattering between selected zones
SAGIT	fictitious scattering within a zone
RFCAVT	RF kick at the boundary of selected zones
EDGEUS	edge scattering
VFAN	define the volumes of all non-Standard zones
MHSETU	booking of user defined histograms
MFILL	filling of user defined histograms
WRTSUR	tag particles which cross selected user defined surfaces
TAGPR	particle origin tagging
BLPROCESS	Block Data of switches to control certain production processes
HISTDUMP	volume and surface histogram binning and DUMP regions
MARS2BML	MARS to Beam-line transformation
BML2MARS	Beam-line to MARS transformation
BLINIT	MMBLB initialization
BLGEOINIT	MMBLB element registration
TUNNELGEO	MMBLB independent tunnel description

5.1 Subroutine MARS1504

Subroutine MARS1504 is the main "steering" routine, which directs the running mode of the executable. There are three possible running modes: normal, GUI interface, and stand-alone event-generator. Switching between these modes is controlled by the **CTRL** card in the input deck. The default mode is normal, where the primary particle interacts with the described geometry, secondary particles are tracked through the geometry, and various parameters are "scored" and tabulated for output. The GUI interface, Section 11, is very useful for checking a complex encoded geometry, and for displaying results. No events are generated in this mode, but the user's encoding of the modeled geometry, and zone and material assignment, is executed and displayed. The stand-alone event generator is useful for ???? .

The three modes can be seen in the 3-way IF-ELSE block within the subroutine. The first segment in the block, which calls STTCL, is the GUI interface mode. The second segment in the block, which calls EVTGEN, is the stand-alone generator mode. The third segment in the block, which calls MARSON, is the normal running mode. For normal and GUI modes, the user does not need to customize anything in routine MARS1504. The only items which might be modified, as needed, are the declared size of the PAWC common block, and the names of the input, output and histogram files. Some of the settings for the stand-alone generator mode are customized from this routine.

5.1.1 Stand-alone event generator

The stand-alone generator, as mentioned above, is useful for ??? To make use of this mode, the user must first set the card **CTRL** 2=IM in the input deck, where IM is the material index of the target nucleus the event generator will use. The primary particle type and energy are specified in the usual way using the **ENRG** and **IPIB** cards in the input deck. For example, if the user wants to investigate the details of protons striking a graphite (carbon) nucleus, then the pertinent cards in the input deck might be

```
CTRL 2=4
NEVT 100
ENRG 250.
IPIB 1 3
NMAT 5
MATR 'AIR' 'AL' 'FE' 'C' 'CONC'
```

where **CTRL** 2=4 indicates that the target nucleus is the 4th material listed in the **MATR** card, which is 'C' for carbon. The primary particle is a 250 GeV proton, and 100 events have been requested.

The user next sets variables in routine MARS1504 to control the generator details and the results which will be reported. The choices are given by comments within the routine: Here, the differentials DY, DMT, X and E are for rapidity, transverse mass, Feynman x, and energy, respectively. Selecting one of these determines the contents of the tables of data which are the result of the calculation.

5.2 Subroutine MIXTUR

Most commonly used compounds for modeling accelerator and detector components are listed in Table 3.4. A compound, or mixture, not listed can be defined by the user via subroutine MIXTUR. First, the user must list his custom mixture(s) in the input deck, as one of the items in the MATR card. The name of the mixture(s) must be MIX1, MIX2, MIX3, etc., up to MIX0 for the first 10; the name MIX0

can continue to be used for additional entries. Each of these names is assigned an `I` index, according to their order in the list of materials.

The user must then know the chemical formula for each mixture. For example, if a target is made from a special alloy, the chemical formula of the alloy is required. From the chemical formula, the user obtains the information necessary for this subroutine: the number of elements in the compound `MIX`, where `MIX` is less than 20, the atomic masses `A(MIX)` of each element, the atomic numbers `Z(MIX)` or each element, and the relative fractions, `W(MIX)`, by weight, of each element in the compound. One can look up `A` and `Z` in the periodic table, but then must calculate the relative weight fractions, using $WMIX_k = (P_k A_k)/AMOL$, where P_k is the total number of k atoms in the compound, and $AMOL = \sum(P_k A_k)$.

The example here shows a case where the 1st three materials are built-in, and the fourth is SiO_2 with 5% of water H_2O by weight. There are thus 1 silicon atoms, 3 oxygen atoms, and 2 hydrogen atoms in the mixture. The input deck would contain a line

```
          NMAT 5
          MATR 'C' 'FE' 'AL' 'MIX1' 'CONC'
```

The users customized `MIXTUR` subroutine is then :

```
      SUBROUTINE MIXTUR(I, MIX, AMIX, ZMIX, WMIX)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
      DIMENSION AMIX(*), ZMIX(*), WMIX(*)
C
      IF(I.EQ.4) THEN
C
C 1=Silicon, 1 atom; 2=Oxygen, 3 atoms; 3=Hydrogen, 2 atoms
      MIX=3
      ZMIX(1)=14.0D0
      ZMIX(2)=8.0D0
      ZMIX(3)=1.0D0
      AMIX(1)=28.0855D0
      AMIX(2)=15.9994D0
      AMIX(3)=1.00794D0
      AMOL1=AMIX(1) + 2.D0*AMIX(2)
      AMOL2=AMIX(2) + 2.D0*AMIX(3)
      AMOL=0.95D0*AMOL1 + 0.05D0*AMOL2
      WMIX(1)=0.95D0*AMIX(1)/AMOL
      WMIX(2)=(0.95D0*2.D0*AMIX(2) + 0.05D0*AMIX(2))/AMOL
      WMIX(3)=0.05D0*2.D0*AMIX(3)/AMOL
C
      END IF
      RETURN
      END
```

On entry to `MIXTUR`, there must be a test on the material index `I`. Exit the routine for all built-in materials, and use a conditional statement to isolate custom mixtures from each other if there are more than one. Remember that all `MARS` variables, such as those passed by subroutine arguments, are double precision.

5.3   Subroutine BEG1

Routine `BEG1` is used to customize the primary particle beam. The subroutine arguments are the variables used to fully describe the trajectory and properties of a given primary particle: `JJ` is the

particle type index given in Table 3.2; W is the relative weight given to the particle; E is the energy of the particle; X,Y,Z is the particle's starting location; DX,DY,DZ are the direction cosines; TOFF is ?; INTA is a flag to impose a point-like interaction ? ; NREG1 is a Zone number where the source term is located ? . When the primary beam is described using Input Deck IPIB and BEAM cards, MARS uses those input deck parameters on beam spot size and shape to generate a particle trajectory within that specified envelope. The trajectory is described by the starting position X,Y,Z and the direction cosines DX,DY,DZ. These variables are passed to BEG1 and can be further modified to describe beam envelopes more complex than what can be specified only by the Input Deck cards.

For example, consider a beam which has a gaussian distribution of particles, but has been clipped in one view. The input deck cards are used to describe the gaussian properties. In routine BEG1 the user examines the X,Y position. If the X or Y are beyond the clipped edge, then the trajectory can be regenerated randomly in BEG1, rechecked, and accepted only when it fits within the users specification. Another example is where the beam is uniformly distributed in one view but gaussian distributed in the other view. The Input Deck cards can do one or the other distribution for both views together, but not separately. Again, the user would use the input deck to specify either the uniform or gaussian characteristics, and then modify those characteristics as needed in routine BEG1.

Question: if you set W=0.0 for primaries which don't fit your criteria, does that eliminate that particular primary particle, and force the generation of a new one?

Another way to use routine BEG1 is to read in a file which contains a list of primary beam particles. The user can create such a file in any custom format; it must simply contain the same parameters as the BEG1 subroutine arguments, or contain sufficient information to extract those parameters. The user in this case must also install the code to properly open, read, and close the file.

Such a file can be generated by MARS itself, using the LEAK subroutine. In this case, one uses MARS in two (or more) consecutive steps. The first step generates a beam into a geometry and creates the resulting showers. Within that geometry is a Zone with a specific Zone number which acts as a flag for subroutine LEAK - any particle crossing into that zone will have it's parameters written to a file. The parameters are exactly those which are the BEG1 subroutine arguments. The first step process ends, resulting in this file of particles which reached that particular Zone. A second MARS process is started, consisting of the geometry "downstream" of the first process's LEAK zone; the file of particles from step 1 is the input for routine BEG1 in step 2. This two or multi-step process is useful when modeling labyrinths or shielding penetrations: step 1 generates many particles, a few of which reach the entrance of a labyrinth (the LEAK zone) and are written to output; step 2 works only with the particles which enter the labyrinth, and models only the labyrinth portion of the geometry. In this case different energy thresholds can be applied in steps 1 and 2 in order to optimize execution time, for example to concentrate only on thermal neutrons in the labyrinth. The multi-step process is also useful when investigating the effects of beam loss distributions: step 1 models the actual beam loss and records the particles which are created in that interaction shower; step 2 tracks only the shower to investigate it's effects on the surrounding area. Another example if the LEAK - BEG1 multi-step process is to separate the primary event generator from the processes which occur in the rest of the geometry: model a primary beam hitting a target, use DPMJET or ISAJET to model what happens in the target, and record, via LEAK, everything which exits the target; input the resulting particle list into step 2, which describes the geometry downstream of the target. In this way one can compare different production models at the target separately from the rest of the geometry.

5.4 Subroutines REG1 and VFAN

REG1 is the interface between MARS and the users's encoding of a complex geometry. MARS hands to the routine, via the subroutine arguments, the current location X,Y,Z of some particle; the user's job is to determine what particular volume that X,Y,Z belongs to, and what the corresponding user-assigned non-Standard zone number N for that volume is, and return that zone number to MARS. Subroutine VFAN is used to assign a volume to each of the user-assigned zones at program start-up. Recall, from Section 3.1, that for MARS to determine the flux of particles through any particular zone, it must have a value for the volume of the zone.

The user can encode as complex a geometry as necessary, and can do so in his own set of subroutines which get called from REG1. The user has as many as 100,000 non-Standard zones at his disposal; each declared zone gets the full complement of results accumulated, and these are tabulated in the MTUPLE.NON output file. Therefore, the boundaries of the user's non-Standard zones will correspond to various regions of interest where specific results are desired. For example, two separate zones of soil on either side of a beam dump can be set up, so that one can compare the star densities in the two locations. Or, set up several zones within an irregularly shaped target holder assembly, so that energy and heat deposition patterns resulting from the beam-target interaction can be studied. There are of course a few requirements which must be met for the proper use of the subroutine. The non-Standard zones must be enclosed within, or have boundaries equal to, the outermost Standard zones. Each zone, of any type, can hold one and only one material. Each zone must have a value for it's volume. The initializations concerning the material content and volumes of the non-Standard zones are performed on the first call to routine REG1.

A simple example will serve to illustrate the non-Standard zone initialization process, as well as the geometry encoding process. It is strongly suggested that the user create a conceptual sketch of his geometry before starting to encode it, filling in the zone boundary locations, materials, zone numbers, and zone volumes, as this will assist in the coding process. Once a geometry is encoded, the user is advised to utilize the MARS GUI visual interface to check that the zone boundaries, and the zone number and material assignments, match the conceptual sketch. Figure 20 shows a sketch of a simple beam dump, made from steel and concrete blocks. These rectagular shapes become non-Standard zones. The dump is surrounded by soil, and the outer edge of the soil is modeled as a cylinder; this can therefore be a Standard zone specified by cards in the input deck (this cylinder is not shown in the sketch).

The following section of code is the array and variable declaration section from subroutine reg1 corresponding to this example of a simple beam dump. The user has decided that material index 1 is steel, material index 2 is concrete, and material index 3 is soil. Consequently, the material cards in the input deck are set to

```
NMAT 3
MATR 'FE' 'CONC' 'SOIL'
```

The user has set the maximum number of non-Standard zones to 20. The first 10 non-Standard zones will be used for steel, and the second 10 for concrete.

```
      SUBROUTINE REG1(X, Y, Z, N, NIM)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C
      INCLUDE 'blreg1.inc'
      INCLUDE 'tally1.inc'
C
      SAVE NENTER
```
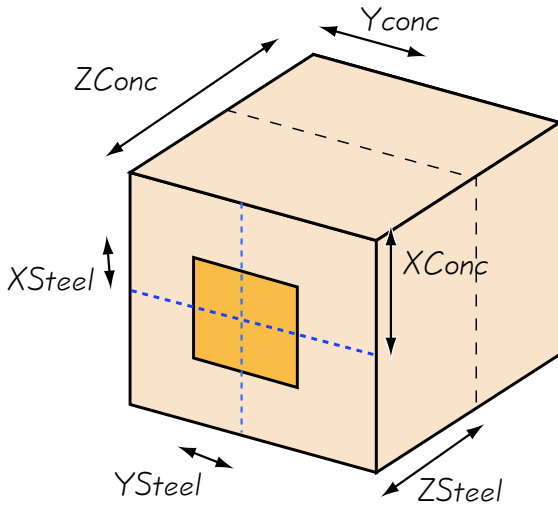
Figure 20: A user's sketch of a simple beam dump, with dimension variables, and with preliminary non-Standard zone assignments.

```
      DATA NENTER/0/
C
C ===Put actual max local non-standard zone number here !!!
      PARAMETER (M\_MAX = 20)
      PARAMETER (M\_MAX1 = M\_MAX + 1)     !+++Don't touch !!!
C
      CHARACTER*8 VNAME, VNAMEBUF
      DATA VNAMEBUF/'        '/
C
      DIMENSION IMUN(1:M\_MAX1)   ! local array for material indicies
      DATA INCREM/1/
      DATA IMUN/ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
     >           2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
     >           0/
```

The local array `IMUN` is declared and sized to `M_MAX + 1`. The contents of `IMUN` are the materials assigned to each of the Mth non-Standard zones, with the value being the index of the material, given by it's order in the input deck list. In other words, `IMUN(M)` is the material index for the Mth zone, where $M = 1$: `M_MAX` - a local zone number. The actual zone number, in the MARS program array space, depends on how many Standard and Extended Geometry zones have also been declared. The following section of subroutine `REG1` initialization code demonstrates this.

```
        IF(NENTER.EQ.0) THEN
           CALL REG3
           NCELMX = NFZPEX + M_MAX
           NENTER = 1
           IF(M_MAX .EQ. 0) INCREM=-1
           IF(M_MAX .GT. 0) THEN
              INUG = 1
              WRITE(*,*)'There are non-standard zones M_MAX= ',M_MAX
              DO L = 1, M_MAX, INCREM
                 MATIND(NFZPEX + L) = IMUN(L)
                 VOLNM (NFZPEX + L) = VNAMEBUF
```

73

```
            END DO
            NVTEST = 1
            CALL VFAN (NVTEST,V)
        ELSE
            INUG = 0
            WRITE(*,*)'There are no non-standard zones in this run !'
            RETURN
        END IF
    END IF
```

The non-Standard zone initialization process is one of the few places where the user directly sets variables held in a MARS program common block. Variable NCELMX, and others, are passed between routine REG1 and the rest of MARS via blreg1.inc, which contains various MARS parameters and common blocks. The line NCELMX = NFZPEX + M_MAX is how MARS is informed of the number of declared non-Standard zones. The total number of all zones, Standard, Extended and non-Standard, is held in variable NCELMX. All of the arrays which hold accumulated MARS results are one-dimensional and indexed by NCELMX. The one-dimensional arrays are logically divided into three blocks, holding first the results for Standard zones, then Extended, then non-Standard. The boundaries of these three blocks are defined by additional variables which are the total numbers of zones of each type. The total number of Standard zones is held in variable NFZP; the total number of Extended zones is in NEXG2; the total of Standard and Extended zones together is given by variable NFZPEX. MARS calculates NFZP, NEXG2, and NFZPEX during its startup initialization, based upon the declarations made in the MARS.INP and GEOM.INP input decks. On the first call to routine REG1, the user declares the total number of all zones, and the number of non-Standard zones, by setting NCELMX.

Notice that the local IMUN array, where materials are assigned to non-Standard zones, is copied to MARS array MATIND, offset in that array's space by NFZPEX. Array MATIND is what MARS uses to determine the material content of each zone, not IMUN. However, it is easiest for the user to think of his own non-Standard zone assignments as being numbered $1 : $ M_MAX, and fill his own local version of the materials array, since the absolute zone number which MARS uses can change if the user changes the number of Standard or Extended zones in his model from one execution of the program to another. All that MARS requires is that the user's version of the materials array be transfered to the MATIND array, with the correct offset, at some point in the REG1 initialization process.

Before continuing with the zone initialization process description, jump ahead to the geometry definition code within REG1 for the simple beam dump example. The user makes a list of parameters to hold the dimension values of his model (remember the basic length unit is centimeters, and all variables double precession) -

```
    Parameter (ZConc = 400.0D0)
    Parameter (ZSteel = 150.0D0)
    Parameter (ZSteel_middle = 75.0D0)
    Parameter (XSteel = 30.0D0)
    Parameter (YSteel = 35.0D0)
    Parameter (XConc = 100.0D0)
    Parameter (YConc = 120.0D0)
```

The encoding of the simple beam dump is then -

```
    M=0
    AX = Abs(X)
    AY = Abs(Y)
C
C Return immediately if we are outside the User non-Standard geometry
```

```fortran
      If (AX.ge.XConc .or. AY.ge.YConc .or. Z.ge.ZConc) Return
C
C Is the current position in the steel core?
      If (AX.lt.XSteel .and. AY.lt.YSteel .and. Z.lt.ZSteel) Then
          If (Z.ge.0.0D0 .and. Z.lt.ZSteel_middle) Then
            M = 1
            Go to 200
          Else If (Z.ge.ASteel_middle .and. Z.lt.ZSteel) Then
            M = 2
            Go to 200
          End If
      End If
C
C Is the current position in the concrete surrounding the steel?
      If (Z .lt. ZSteel) Then
          If (AX.lt.XConc .and. (Ay.ge.YSteel .and. AY.lt.YConc)) Then
            M = 11
            Go to 200
          Else If ((AX.ge.XSteel .and. AX.lt.XConc) .and. AY.lt.YConc) Then
            M = 11
            Go to 200
          End If
C Is the current position in the concrete downstream of the steel?
      Else If (Z.ge.ZSteel .and. Z.lt.ZConc) Then
          If (AX.lt.XConc .and. AY.lt.YConc) Then
            M = 13
            Go to 200
          End If
      End If
C
C If all User-zone-area X,Y,Z are covered by the above statements, then the
C code should never get to this point. If M=0 here, then there is a hole in the
C logic. If M>0 and gets to here, then there is a missing Go To 200 statement
C Print a warning
      write (*,*) ' **** Logical Error in Reg1 **** ',M
      Return
C
C Set and Return N, the Mars Zone number corresponding to input X,Y,Z
 200  Continue
      IF (M .GT. 0) THEN
          N = NFZPEX + M
      ELSE IF (M .LT. 0) THEN        ! Non-standard blackhole
          N = M
      END IF
      RETURN
      END
```

Although this code is simple, there are several necessary steps which apply to any use of this subroutine. When REG1 is called, MARS passes to it the position X, Y, Z of the current particle. There is a local variable M which is set to 0 on entry to REG1; M is set to a non-zero value somewhere in the code's logic, and it's value is between 1 and M_MAX. Just before exiting REG1, the statement N = NFZPEX + M sets and returns to MARS the absolute zone number N which corresponds to the input X, Y, Z.

There are other steps in this example which, while not necessary, are highly recommended. At the top, there is a statement (or series of statements) which asks whether the current X, Y, Z is within the regions where non-Standard zones are defined; if not, then the routine is exited. Up to 80% of

the program's running time is spent in geometry modules, including REG1. It is far more efficient to determine immediately whether X, Y, Z is outside the overall non-Standard zone region, than to let all possible X, Y, Z locations fall through the code. In the case where X, Y, Z is outside the non-Standard zone region, the input absolute zone number N must be returned to MARS **unchanged**. Another recommendation is to construct the logic of the code so that satisfied conditions 'jump out', and errors 'fall-through'. In this example, once a condition is satisfied, and a zone number assigned, the code 'jumps out' of the conditional statements to the end, where the absolute zone number N is assigned. If none of the conditional statements are satisfied, the code 'falls through' to the end of this list of statements. If the conditional statements are coded correctly, then the code should never reach the 'fall through' point. One can use this fact to catch errors. While this tactic may seem unnecessary for a simple example, the logic to encode more intricate geometries can rapidly become very complex, and difficult to debug.

Now that the encoding and zone assignment of the simple beam dump is defined, return to the zone initialization process. Every non-Standard zone where the user wants results to be tabulated must have it's volume declared to MARS. The default method for handing volume values to MARS is to use subroutine VFAN. As shown in the above code section, the default version of REG1 is set up to call VFAN in it's initialization section, and similarly, VFAN contains an initialization section of it's own, executed this first time the routine is called. For the simple beam dump example, subroutine VFAN is as follows -

```
      SUBROUTINE VFAN (N, V)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C
      INCLUDE 'blreg1.inc'
      INCLUDE 'tally1.inc'
C
      DATA NENTER/0/
      SAVE NENTER
C
      Parameter (ZConc = 400.0D0)
      Parameter (ZSteel = 150.0D0)
      Parameter (ZSteel_middle = 75.0D0)
      Parameter (XSteel = 30.0D0)
      Parameter (YSteel = 35.0D0)
      Parameter (XConc = 100.0D0)
      Parameter (YConc = 120.0D0)
C
C- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      N1=NFZPEX    ! NUMBER OF STANDARD+EXTENDED REGIONS
C
C one-time initialization section
      IF(NENTER.EQ.0) THEN
         NENTER=1
C
C Steel core - both sections the same volume
         VOLUME (N1 + 1) = (2.0D0*XSteel)*(2.0D0*YSteel)*ZSteel_middle
         VOLNM (N1 + 1) = 'UpstCore'
         VOLUME (N1 + 2) = Volume (N1 + 1)
         VOLNM (N1 + 2) = 'DnstCore'
C
C The concrete around the sides of the steel core
         A1 = 2.0D0*((2.0D0*YConc)*(XConc-XSteel))      ! Top and Bottom area
         A2 = 2.0D0*((YConc-YSteel)*(2.0D0*XSteel))     ! two sides area
         VOLUME (N1 + 11) = (A1 + A2) * ZSteel
         VOLNM (N1 + 11) = 'Conc Zn1'
C
```

```
C The concrete in back of the steel core
        VOLUME (N1 + 13) = (2.0D0*YConc) * (2.0D0*XConc) * (ZConc - ZSteel)
        VOLNM (N1 + 13) = 'Conc Zn2'
C
        Return
C
C non-initialization call - return the Volume for Zone N
      Else
         V = VOLUME (N)
      End If
C
      RETURN
      END
```

The same parameter list, used in `REG1`, to describe the dimensions of the beam dump, is also included here to calculate the volumes of the beam dump zones; all volumes are cm^3. Zone volumes are placed directly into the MARS array `VOLUME`; these non-Standard zone volumes are written with an array offset of `NFZPEX`, just as the material content of each non-Standard zone was entered into array `MATIND`. At program startup, MARS initializes all elements of array `VOLUME` to zero. MARS then fills the Standard and Extended zone sections of array `VOLUME` based on settings in the `MARS.INP` and `GEOM.INP` input files. Finally, the user's code is called to fill the non-Standard zone portion.

Also filled is character array `VOLNM`, which holds an 8-character string name for each zone. This array is initialized to hold blanks in the initialization section of subroutine `REG1`. In subroutine `VFAN` this array gets filled with zone names for each zone which gets a volume assigned. These zone names are used in the non-Standard zone output file `MTUPLE.NON`, and simply make it easier to locate the line holding the results for a specific zone.

The non-Standard output file `MTUPLE.NON` holds a table of results for all declared non-Standard zones. Only zones which have been properly declared will have an entry in the table of results, and the trigger for what MARS considers proper declaration is a non-zero value for the volume of the zone. So, even if there is a material indicated for the zone entered into the `MATIND` array, even if code exists within `REG1` (or called from it), which describes the boundaries of the zone, and this code returns a non-Standard zone number assignment, no results for that zone will be reported in the output unless the zone has a non-zero volume entered into array `VOLUME`. And, for the results to be correct, the volume value cannot be arbitrary - it must be the correct volume, corresponding to the encoded boundaries of the zone. As discussed in section 3, that is is particle path length which is accumulated as particles are tracked through zones, and to obtain the particle flux in the zone, path length is divided by the zone volume. An incorrect value for the zone volume will give incorrect results for the flux, and for all other results derived from particle fluxes.

The same local zone index is used when defining volumes.

can be greater than the total number of non-Standard zones actually used, because the declared array space does not need to be close-packed.

M_MAX is set by the user, and is the size of the MARS array space being reserved for the user's non-Standard zones. The user can define M_MAX in a `DATA` statement, as done in the distributed `REG1`, or via his own parameter list or common block.

The user must be careful to coordinate the physical outer boundaries of his non-Standard volume with the Standard zone(s) and/or Extended geometry zones defined via the input decks. Non-Standard zones must all fit inside Standard zones, or be aligned along the same boundaries; non-Standard zones cannot extend beyond the outermost defined Standard zone. This is because, as stated above, the

outermost extent of the Standard zones defines for MARS the outermost extent of the entire model. If the entire model is defined by the user from REG1, then one must still have at least 1 Standard zone, and the maximum geometrical values in the ZSEC and RSEC cards in the input deck must correspond to the maximum geometrical limits set by the user within REG1. Up to 80% of the cpu-time is spent in geometry modules, including REG1, when propagating particles. If the user's non-Standard zones fit within larger Standard zones, then the user should test the input X,Y,Z on entry to REG1, and if the current location is outside the overall non-Standard volume, RETURN to MARS, with the input zone number N unchanged. This is far more efficient than letting all possible X,Y,Z locations fall through the user's code.

If one wants to study cascades in a very complex geometry not embraced by the above options, user subroutines REG1 and REG2 must be provided. MARS, version 13(95), allows the user to place geometrical objects of almost any complexity inside the pre-defined $standard$ (r-z-ϕ) or $extended$ geometry. By convention the total number of $standard$ regions NFZP must be NFZP$\geq$1. With the help of his/her own subroutines REG1(X,Y,Z,N,NIM) and REG2(N,IM,MAG) the user can describe arbitrary physical regions numbered from the NMIN$\leq$N$\leq$NMAX interval. Here NMAX=10000; NMIN=NFZP+1 for the Standard geometry sector and NMIN=NEXGM+NVOLUM for the $extended$ geometry sector. Each region can be divided into any number of arbitrary subregions with index NIM with a default value NIM=0. This feature provides the possibility of distinguishing geometrical zones without scoring results there. The user can successfuly substitute this with the HBOOK histogramming (see MAIN).

For each call, a subroutine REG1(X,Y,Z,N,NIM) finds the position of the given point (X,Y,Z) in the system: it determines the corresponding physical region number N (each with its own material index) and, if one wishes, the subregion number NIM. The last parameter can be used in a subroutine FIELD to determine the type of magnetic field (uniform, dipole, quadrupole etc.) in the region N. Default: MAG=0 (no magnetic field in the region).

By convention the region outside of the global volume has a number N=0 and properties of the $black\ hole$. In some applications it is useful to tag and to score the leakage out of the non-standard regions into the $black\ holes$ labeled with N$\leq$ -1 to use these negative tags in a user routine LEAK. The user must pay a special attention to careful programming of the subroutine REG1, because the geometrical modules consume usually about 80 % of the CPU time, as is typical of cascade Monte Carlo programs.

A simple example of subroutine REG1 is:

```
      SUBROUTINE REG1(X,Y,Z,N,NIM)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  NON-STANDARD GEOMETRY MODULE
C  FINDS THE PLACE OF GIVEN POINT IN THE SYSTEM
C  INPUT: X, Y, Z
C  OUTPUT:
C  N - PHYSICAL REGION NUMBER, NMIN≤N≤NMAX
C  NMIN=NFZP+1 (STANDARD)
C  OR
C  NMIN=NEXGM+NVOLUM (EXTENDED)
C  NMAX=10000
C  N≤-1 DEFINES NUMBERED LEAKAGE OUT OF THE SYSTEM
C  IN NON-STANDARD SECTOR
C  NIM -GEOMETRICAL SUBREGION NUMBER, 0≤NIM
C
C  REVISION: 28-FEB-1995
C
      R=SQRT(X*X+Y*Y)
      IF(R.LT.10.)  RETURN
      M=500
      IF(R.LT.150.)  THEN
      IF(Z.LT.490.)  THEN
C  STEEL SHIELDING:
      N=M+1
C  TUNNEL:
      IF(R.GT.30.)  N=M+2
      END IF
      ELSE
C  CONCRETE SHELL:
      N=M+3
      END IF
      RETURN
      END
```

5.5 Geometries (REG3)

In some cases it is convenient to re-define a few material and/or magnetic indices assigned to the *standard* or *extended* regions at the initilization stage. This can be easily done in a user routine REG3, e.g.:

```
      SUBROUTINE REG3
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  RE-DEFINES IM AND MAG FOR STANDARD SECTOR
      IF(N.EQ.137) IM=14
      IF(N.EQ.2503) IM=2
      RETURN
      END
```

5.6 Magnetic and Electrical Fields (FIELD, SUFI, RFCAVT)

To describe the magnetic field components (BX,BY,BZ) in the regions with parameter MAG$\neq$0 the user puts IND(4)=T and provides a subroutine FIELD. The same routine is used to describe an electrical field. One can use a corresponding map or analytical expressions to find the field components in the point (X,Y,Z). Parameter MAG defined in REG2 can speed up the search. It indicates the type of the field in the region. The unit for magnetic field is $Tesla$. A 2-D or 3-D field map is read in a user routine SUFI at the initilization stage and transfered to the routine FIELD via appropriate COMMON block.

An example of the FIELD subroutine is:

```
      SUBROUTINE FIELD(N,X,Y,Z,BX,BY,BZ,BBB)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  FINDS COMPONENTS OF MAGNETIC FIELD
C  INPUT: MAG -MAGNETIC INDEX AT GIVEN POINT
C  X,Y,Z - POINT'S COORDINATES (OPTIONAL)
C  FIELD MAPS
C  QUADS GRADIENTS IN T/CM
C  G>0 FOR FOC QUADS, G<0 FOR DEFOC QUADS
C  OUTPUT: BX,BY,BZ,BBB IN TESLA
C  REVISION: 16-DEC-1994
C
      COMMON/BLINT2/JJ,KK,MAG
      DATA RAQ,BQUA,G1/8.,2.,0.53/
      BX=0.
      BY=0.
      BZ=0.
      IF(Z.LT.622.)  THEN
      BX=5.
      IF(MAG.EQ.2) BX=-5.
      ELSE
      CALL QUAD(X,Y,R,RAQ,G1,BX,BY)
      CALL DIPOLE(X,Y,R,RAQ,BQUA,BX1,BY1)
      BX=BX+BX1
      BY=BY+BY1
      END IF
      RETURN
      END
```

```
      SUBROUTINE RFCAVT(JJ,VECT,VOUT,E,TOFF,NIB,NIM,IFLAG)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  RF CAVITY KICK:
C  DISCRETE AT BOUNDARIES BETWEEN REGIONS WITH SPECIAL NUMBERS NIB AND NIM
C  NIB - PRIOR CROSSING, NIM - AFTER CROSSING
C  INPUT : JJ, VECT (X,Y,Z,DCX,DCY,DCZ,P), E, TOFF, NIB, NIM
C  OUTPUT: VOUT (X,Y,Z,DCX,DCY,DCZ,P), E, IFLAG
C  VECT(1)-VECT(7) AND E ARE RE-DEFINED TO VOUT(1)-VOUT(7) AND E
C  IFLAG=1 MUST BE RAISED
C  IFLAG=2 - $E_{new} < 0$
      INCLUDE 'cmasnsg.inc'
      DIMENSION VECT(7),VOUT(7)
C  DO L=1,7
C  VOUT(L)=VECT(L)
C  END DO
      RETURN
      END
```

5.7 Fictitious Scattering (ALIGN, SAGIT)

In some applications components of the considered system can be turned or shifted with respect to each other. Say, the arc of any circular accelerator is built of the magnets turned by a fixed angle. MARS, version 13(95), allows an elegant way to handle such systems. The user describes the geometry as the straight along the z-axis. Then, with the help of a routine ALIGN, he/she creates the angular or space kicks at the required boundaries in the directions *opposite* to the real ones. One can easily see that the resulting coordinates and angles of any particle are identical to those in the real *bent* geometry. It is convenient to use spare NIM parameters to control such "a fictitious scattering" at the required boundaries with NIB≠NIM as that parameter in a previous region. Of course, any coordinate can be used to locate the needed boundary.

The following example shows use of the ALIGN routine for a single kick:

```
      SUBROUTINE ALIGN(VECT,VOUT,NIB,NIM,IFLAG)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  FICTITIOUS SCATTERING:
C  DISCRETE AT BOUNDARIES BETWEEN REGIONS WITH SPECIAL NUMBERS NIB AND NIM
C  NIB - PRIOR CROSSING, NIM - AFTER CROSSING
C  VECT,VOUT: X,Y,Z,DCX,DCY,DCZ,P
C  VECT(1)-VECT(6) CAN BE RE-DEFINED TO VOUT(1)-VOUT(6),
C  NORMALLY AT IND(4)=T
C  IF SO, VOUT(1)-VOUT(6) MUST BE FILLED AND IFLAG=1 MUST BE RAISED
      DIMENSION VECT(7),VOUT(7)
C  DO L=1,6
C  VOUT(L)=VECT(L)
C  END DO
      RETURN
      END
```

```
      SUBROUTINE SAGIT(CHARGE,STEP,VECT,VOUT,NREG,IFLAG)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  FICTITIOUS SCATTERING:
C  ON STEP IN REGION NUMBER NREG
C  VECT,VOUT: X,Y,Z,DCX,DCY,DCZ,P
C  VECT(1)-VECT(6) CAN BE RE-DEFINED TO VOUT(1)-VOUT(6),
C  NORMALLY AT IND(4)=T
C  IF SO, VOUT(1)-VOUT(6) MUST BE FILLED AND IFLAG=1 MUST BE RAISED
      DIMENSION VECT(7),VOUT(7)
C  DO L=1,6
C  VOUT(L)=VECT(L)
C  END DO
      RETURN
      END
```

The same method is used in MARS to handle objects with saggita, i.e. continuously bent. "Fictitious scattering" defined in a user subroutine SAGIT as an *anti-kick* at every step along charged and neutral particle trajectories, simplifies the geometry description and makes a precise correspondence to the real bent objects.

5.8 Edge Scattering (EDGEUS)

If the "edge scattering problem" is considered with IND(9)=T and the geometry includes $non-standard$ insertions defined with REG1 and REG2, the user should supply a subroutine EDGEUS(X,Y,Z,DX,DY,DZ,U,V). It finds the distance U from the given trajectory point (X,Y,Z) to a non-standard surface, and the projection V of the direction vector (DX,DY,DZ) to the normal to the surface which passes in its positive direction through the point (X,Y,Z).

This example shows use of the EDGEUS routine for the surface Y=YPL>0 and for the particle with Y>YPL and DY<0:

```
      SUBROUTINE EDGEUS(X,Y,Z,DX,DY,DZ,U,V)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  EDGE-SCATTERING PROBLEM
      U=Y-YPLANE
      V=DY
      RETURN
      END
```

5.9 Leakage (LEAK)

A user subroutine LEAK(N,K,JJ,W,E,X,Y,Z,DX,DY,DZ) handles particles which escape from the system (N=0) or from the $non-standard$ regions tagged with N≤ -1. Parameter K is the tree vertex level (generation number) for the hadronic part of the calculated cascade. If IND(14)=T, then K has the same meaning as the above if given particle was generated in hA vertex at E>0.0145 GeV, otherwise it is forced to be K=−205, if the leaked neutron was produced in sequent interactions below 0.0145 GeV. JJ is particle type from Table 3.2. If K=−205, the particle type is neutron by default and JJ is its energy group number from Tables **??** or 6.4. W is leaked particle statistical weight, E is its kinetic energy, (X,Y,Z) and (DX,DY,DZ) - its coordinates and direction cosines, respectively.

The routine LEAK used in an example in Section 5.17 to collect generated antiprotons in the file PBAR.OUT looks like:

```
      SUBROUTINE LEAK(N,K,JJ,W,E,X,Y,Z,DCX,DCY,DCZ,TOFF)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  PARTICLE LEAKAGE SPECIAL SCORING
C  IF(K=-205):  JJ=NGROUP, E=0.0005*(EN(JJ)+EN(JJ+1))
C  FOR L.E.NEUTRONS (E≤0.0145 GEV)
C  REVISION: 02-NOV-1994
      COMMON/STAZI1/ZMAX,REXT
      DATA POPT,DP0,DZ0/8.9,0.022,0.99/
C
      IF(JJ.NE.12) RETURN
      IF(DZ.LE.DZ0) RETURN
      P=SQRT(E*(E+1.87656))
      DP=ABS((P-POPT)/POPT)
      IF(DP.GT.DP0) RETURN
      WRITE(9)E,W,X,Y,Z,DX,DY,DZ,K
      RETURN
      END
```

5.10 Special Volumes (VFAN)

In a *standard* geometry sector the volumes of the regions needed, for example, to compute energy deposition density in GeV/g, are calculated at the final stage in the SERVN subroutine. The array of volumes VV of the $non-standard$ regions M should be provided by the user in a subroutine VFAN(N,V). The same subroutine can be used in specific activity and residual dose rate calculations at IND(13)=T. The VFAN routine can look as:

```
      SUBROUTINE VFAN(N,V)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  FIND VOLUME V(N),CM**3 OF REGION N OF THE NON-STANDARD GEOMETRY
C  VOLUME(N) ARE DEFINED IN SERV FOR N <= NFZP
      INCLUDE 'tally1.inc'
      DATA NENTER/0/
      SAVE NENTER
      N1=NFZPEX (NUMBER OF STANDARD+EXTENDED REGIONS)
      IF(NENTER.EQ.0) THEN
      NENTER=1
C  PUT HERE THE REAL VOLUMES (CM**3) FOR ALL THE NEEDED ===
C  NON-STANDARD REGIONS <= M_MAX, I.E. REDEFINE ANY OF THE
C  PRE-DEFINED VOLUME(L)=0.D0
C  For example:
C  VOLUME(N1+2) =PI*RCOL*RCOL*180.D0
C  VOLUME(N1+3) =PI*(400.D0-RCOL*RCOL)*60.D0
C  VOLUME(N1+365)=VOLUME(N1+3)
      END IF
      V=VOLUME(N)
      RETURN
      END
```

5.11 User Histogramming (MHSETU, MFILL)

```
    SUBROUTINE MHSETU
C  SET UP HISTOGRAM ARRAYS
C  HISTOGRAM ENTRY FOR USER-DEFINED HISTOGRAMMING
C  HISTOGRAM ID AVAILABLE: 700<ID<999
C  HISTOGRAM TYPE:
C  IHTYP = 1 - COLLISION
C  IHTYP = 2 - STEP
C  IHTYP = 3 - ENERGY DEPOSITION
    IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  REMEMBER, HBOOK IS A SINGLE PRECISION ENGINE
C  DON'T FORGET THE 'REAL' DECLARATIONS SUCH AS:
C  REAL AA,ELB,X1,Y1,Y1,Y2
C  CALL HBOOKB(ID,AA,NEB,ELB,0.)
C  CALL HBOOK2(ID,TITLE,NX,X1,Y1,NY,Y1,Y2,0.)
    RETURN
    END
```

```
      SUBROUTINE MFILL(IHTYP,NREG,IM,JJ,E1,E2,DELE,W,X1,Y1,Z1,X2,Y2,Z2,DCX,DCY,DCZ,STEP,TO
C HISTOGRAM ENTRY FOR USER-DEFINED HISTOGRAMMING
C HISTOGRAM ID AVAILABLE: 700<ID<999
C CALL TYPE:
C IHTYP = 1 - COLLISION
C IHTYP = 2 - STEP ("TRACK-LENGTH")
C IHTYP = 3 - ENERGY DEPOSITION (LOCAL OR ON THE STEP)
C NREG - REGION NUMBER FOR COLLISION OR STEP START
C STEP - STEP (cm)
C E1 - ENERGY BEFORE STEP (GeV)
C E2 - ENERGY AFTER STEP (GeV)
C DELE - ENERGY DEPOSITED (GeV), LOCALLY OR ON THE STEP
C W - STATISTICAL WEIGHT
C X1,Y1,Z1 - COORDINATES AT THE STEP START (cm)
C X2,Y2,Z2 - COORDINATES AT THE STEP END (cm)
C DCX,DCY,DCZ - DIRECTION COSINES AT THE STEP START
C TOF - TIME-OF-FLIGHT AT THE STEP END (sec)
C NI - HISTORY NUMBER
C IHTYP=1:  E1=E2, DELE=0, STEP=0, X1=X2, Y1=Y2, Z1=Z2
C IHTYP=3:  E1.ne.E2, DELE>0; STEP=0 (X1=X2, Y1=Y2, Z1=Z2) OR STEP>0 (X1.ne.X2, Y1.ne.
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C REMEMBER, HBOOK IS A SINGLE PRECISION ENGINE
C DON'T FORGET CONVERSIONS OF THE FOLLOWING TYPE:
C REAL EEH,WWH,XL,YL,WH
C EEH=REAL(E1)
C WWH=REAL(W)
C CALL HFILL(ID,EEH,0.,WWH)
C CALL HF2(ID,XL,YL,WH)
      RETURN
      END
```

5.12 Surface Detector Files (WRTSUR)

```fortran
      SUBROUTINE WRTSUR(IUNIT,NI,JJ,E,W,X,Y,Z,DCX,DCY,DCZ,TOFF)
C     PARTICLE AT THE SURFACE DETECTOR
C     FOR WRITING, HISTOGRAMMING ETC.
C     UNIT=81-90 WRITE: fort.81 - fort.90 RESERVED FOR SURFACE DETECTOR FILES
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
      INCLUDE 'cmasnsg.inc'
      PARAMETER (CLIGHT=29979245800.D0)
      PARAMETER (IWRTYP=1) !  momentum components (particle production)
C     PARAMETER (IWRTYP=2) !  energy and cosines (source term for staging)
      IF(IWRTYP.EQ.1) THEN
      ET=E+PM(JJ)
      PA=SQRT(E*(E+2.D0*PM(JJ)))
      CTOFF=CLIGHT*TOFF
      PX=PA*DCX
      PY=PA*DCY
      PZ=PA*DCZ
      WRITE(IUNIT,101)NI,JJ,X,Y,Z,PX,PY,PZ,ET,CTOFF,W
 101  FORMAT(I8,I3,3F11.3,6(1PE14.6))
      ELSE IF(IWRTYP.EQ.2) THEN
      WRITE(IUNIT,102)NI,JJ,E,W,X,Y,Z,DCX,DCY,DCZ,TOFF
 102  FORMAT(I8,I3,5(1PE13.5),3F14.10,1PE13.5)
      END IF
      RETURN
      END
```

5.13 Particle Origin Tagging (TAGPR)

Particle origin tagging for muon and hadron track-length and energy deposition in detector (region number $NREG$ and material index IM) Particle JJ of weight W and energy $E1$ makes a step ALA in that region with energy $E2$ at the step end. It was originated by particle $IORIG$ with energy $EORIG$ and weight $WORIG$ in the process $KORIG$ at the point ($XORIG, YORIG, ZORIG$) in region $NRORIG$ with material index $IMORIG$.

```fortran
      SUBROUTINE TAGPR(NREG,IM,JJ,W,E1,E2,ALA)
C.................................................
*     Particle origin tagging for muon and hadron track-length and energy deposition
*     in detector (region number NREG and material index IM)
*     Particle JJ of weight W and energy E1 makes a step ALA in that region
*     with energy E2 at the step end.
*     It was originated by particle IORIG with energy EORIG and weight WORIG in
*     the process KORIG at the point (XORIG,YORIG,ZORIG) in region NRORIG with
*     material index IMORIG.
*
*     IORIG = IKORIG(1,...) - projectile ID
*     KORIG = IKORIG(2,...) - origin ID
*     NRORIG= IKORIG(3,...) - origin region number
*     IMORIG= IKORIG(4,...) - origin material index
*
*     KORIG = 0 - primary beam
*             1 - muons,   pi/K-decay
```

```
*              2 - muons,   prompt at hA-vertex
*              3 - muons,   Bethe-Heitler pair
*              4 - muons,   e+e- annihilation
*              5 - hadrons, hA-vertex
*              6 - hadrons, elastic
*              7 - hadrons, from muons
*              8 - hadrons, pi/K-decay
*              9 - hadrons, EMS
*             10 - hadrons, recoil LEN
*             11 - hadrons, from neutrinos
*             12 - photons, pi0-decay
*             13 - photons, synchrotron
*             14 - g,e+,e-, at hA vertex
*             15 - e+,e-,   knock-on and prompt e+e- by hadrons
*             16 - e+,e-,   mu-decay
*             17 - e+e-,    prompt e+e- by muons
*             18 - photons, brems
*             19 - photons, stopped muons
*             20 - photons, LEN


C-----
C     CREATED:     11-AUG-2003 BY NVM
C     LAST CHANGE: 21-AUG-2003 BY NVM
C-----
C................................................
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)

      COMMON
     & /BPRTAG/XORIG,YORIG,ZORIG,WORIG,EORIG,IORIG,KORIG,NRORIG,IMORIG
     & /HIST/NI,NSTOP,NUPRI,NHIPR
C- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -


C=== Example ===
C      if(nreg.eq.9) then
C        write(1,100)NI,NREG,IM,JJ,W,E1,E2,ALA,
C     &        XORIG,YORIG,ZORIG,WORIG,EORIG,IORIG,KORIG,NRORIG,IMORIG
C 100    format(' NI,NREG,IM,JJ,W,E1,E2,ALA = ',i7,I5,2i3,4(1pe11.4)/
C     &         ' XORIG,YORIG,ZORIG,WORIG,EORIG,IORIG,KORIG,NRORIG,IMORIG'
C     &        ,'= ',5(1pe11.4),2i3,2i5)
C      end if

      RETURN
      END
```

5.14   Biasing Control (BLPROCESS)

```
      BLOCK DATA BLPROCESS
 C  IPRCEM(K) = 0 - exclusive
 C  IPRCEM(K) = 1 - inclusive with a probability PRCEM(K)
 C  IPRCEM(0) - global control
 C  in process.inc :   INTEGER NPRCEM,NPRCHT,NPRCMT,NPRCNT,NPRCHA
 C  in process.inc :   INTEGER IPRCEM,IPRCHT,IPRCMT,IPRCNT,IPRCHA
 C  in process.inc :   DOUBLE PRECISION PRCEM,PRCHT,PRCMT,PRCNT,PRCHA
      INCLUDE 'process.inc'
 C cems
 C 1 2 3
 C set EMS global bias flag:
 C 0 - exclusive
 C 1 - LPB
 C 2 - HALF
 C DATA IPRCEM/0,NPRCEM*0/
      DATA IPRCEM/1,NPRCEM*0/
      DATA PRCEM/NPRCEM*0.D0/
 C hadron-nucleus vertex
      DATA IPRCHA/0,NPRCHA*0/
      DATA PRCHA/NPRCHA*0.D0/
      RETURN
      END
```

5.15   Histogram Binning and DUMP Regions (HISTDUMP)

```
      BLOCK DATA HISTDUMP
C.........................................................
C     BASELINE VOLUME-SURFACE DETECTOR BINNING
C               AND
C     SELECTED DUMP REGION NUMBERS (0 < NBMA < 300)
C     IN ADDITION TO THE STANDARD GEOMETRY SECTOR
C-----
C     CREATED:     21-NOV-2001 BY NVM
C     LAST CHANGE: 14-FEB-2002 BY NVM
C-----
C.........................................................
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)

      COMMON/BHBKRZ/NZBIN(3),NRBIN(3),NXH,NYH
      COMMON/BLDUMP/NBU(300),NBMA

C=== VOLUME R-Z DETECTOR BINNING (at NOB > 0) ===
      DATA NZBIN/200,200,200/   ! Recommended
      DATA NRBIN/120,120,120/   ! Recommended

C=== SURFACE X-Y DETECTOR BINNING (at NSURF > 0) ===
      DATA NXH/150/   ! Recommended
      DATA NYH/150/   ! Recommended

C=== SELECTED DUMP REGION NUMBERS (0 < NBMA < 300) ===
      DATA NBMA/0/
      DATA NBU/300*0/
```

```
C    For example:
C     DATA NBMA/6/
C     DATA NBU/115,116,117,921,1050,4380,
C    &   294*0/
C
      END
```

5.16   MAD-MARS Interface (MARS2BML, BML2MARS, BLINIT, BLGEOINIT, TUNNEL-GEO)

```
      SUBROUTINE MARS2BML(POS,W,BLPOS,BLW)
C......................................................
C    MARS-TO-BEAMLINE TRANSFORMATION
C    CALLED IF IND(13)=T
C
C-----
C    CREATED:     2002 BY IT
C    LAST CHANGE: 22-APR-2003 BY NVM
C-----
C......................................................
      IMPLICIT NONE

      DOUBLE PRECISION POS(3), W(3), BLPOS(3), BLW(3)
      LOGICAL FIRSTCALL
      DATA FIRSTCALL/.TRUE./
      SAVE FIRSTCALL
      INTEGER IT
C- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

C=== EXAMPLE OF REAL WORK ===

*       INCLUDE 'beamOffset.inc'
*       DOUBLE PRECISION MADVEC(3)
*       DOUBLE PRECISION XMAD,YMAD,ZMAD
*       EQUIVALENCE (XMAD,MADVEC(1)),(YMAD,MADVEC(2)),(ZMAD,MADVEC(3))
*       DOUBLE PRECISION aaa
*       DOUBLE PRECISION MADDIR(3)
*       DOUBLE PRECISION WXMAD,WYMAD,WZMAD
*       EQUIVALENCE (WXMAD,MADDIR(1)),(WYMAD,MADDIR(2)),(WZMAD,MADDIR(3))
*
*       call mars2mad(POS(1),POS(2),POS(3),XMAD,YMAD,ZMAD)
*       call mars2mad(W(1),W(2),W(3),WXMAD,WYMAD,WZMAD)
*
*       if(POS(3).GT.NuMI_2_Z0) then
*          call set_current_beamline(2)
*       else
*          call set_current_beamline(1)
*       endif
*       if(MADVEC(3).GE.11894.92D0.AND.MADVEC(3).LE.11894.94D0) THEN
*          call glob2flat(MADVEC,MADDIR)
*       else
*          call glob2flat(MADVEC,MADDIR)
*       endif
*       call mad2mars(XMAD,YMAD,ZMAD,BLPOS(1),BLPOS(2),BLPOS(3))
*       call mad2mars(WXMAD,WYMAD,WZMAD,BLW(1),BLW(2),BLW(3))
*       if(BLPOS(3).GE.931.0626d0) THEN
```

```
*          aaa=1.d0
*        ENDIF

C=== COMMENT THIS MODULE FOR REAL WORK ===
      IF(FIRSTCALL) THEN
          WRITE(*,*)' MARS2BML: IND(13) is active, but transformation'
          WRITE(*,*)' to the beamline coordinate system is not defined.'
          WRITE(*,*)' See example in MARS2BML subroutine!'
          FIRSTCALL=.FALSE.
      ENDIF
      DO IT=1,3
          BLPOS(IT)=POS(IT)
          BLW(IT)=W(IT)
      END DO
C=========================================

      RETURN
      END


      SUBROUTINE BML2MARS(BML_VEC,BML_DIR,MARS_VEC,MARS_DIR)
C..........................................................
C     BEAMLINE-TO-MARS TRANSFORMATION
C     CALLED IF IND(13)=T
C
C-----
C     CREATED:     2002 BY IT
C     LAST CHANGE: 08-NOV-2002 BY NVM
C-----
C..........................................................
      IMPLICIT NONE
      DOUBLE PRECISION BML_VEC(3),BML_DIR(3),MARS_VEC(3),MARS_DIR(3)
      INTEGER IT
C- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

C=== EXAMPLE OF REAL WORK ===

*       DOUBLE PRECISION CBUF(3),DBUF(3)
*       IF(BML_VEC(3).GE.1234.0989) THEN
*          call set_current_beamline(2)
*       ELSE
*          call set_current_beamline(1)
*       ENDIF
*       CALL mars2mad(BML_VEC(1),BML_VEC(2),BML_VEC(3),
*      &     CBUF(1),CBUF(2),CBUF(3))
*       CALL flat2glob(CBUF,DBUF)
*       CALL mad2mars(CBUF(1),CBUF(2),CBUF(3),
*      &     MARS_VEC(1),MARS_VEC(2),MARS_VEC(3))

C=== COMMENT THIS DO-LOOP FOR REAL WORK ===
      DO IT=1,3
          MARS_VEC(IT)=BML_VEC(IT)
          MARS_DIR(IT)=BML_DIR(IT)
      END DO
C=========================================

      RETURN
      END


      SUBROUTINE BLINIT(M_MAX, IMUN)
```

```
C...........................................................
C     MMBLB INITIALIZATION
C     CALLED IF IND(13)=T
C
C-----
C     CREATED:     2001 BY KRIOL
C     MODIFIED:    17-OCT-2003 BY MAK
C     LAST CHANGE: 04-DEC-2003 BY NVM
C-----
C...........................................................
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)

      INTEGER IMUN(*)

      include 'blreg1.inc'
      include 'tally1.inc'

C=== EXAMPLE OF REAL WORK (uncomment and modify * whereever needed) ===

*       INCLUDE 'SIMPLE_TUNNEL.INC'
*       INITZONE= 1
*       CALL BEAMLINES_IN_USE(2)  ! EXTENSION TO  SEVERAL BEAMLINES
*       CALL SET_CURRENT_BEAMLINE(1)

c= Angle of rotation (azimuth) about vertical axis (MARS X-axis), between
c= Z axis and projection of the beamline direction vector onto (Z,Y) plane.
c= A positive angle forms a right hand screw with MARS X-axis.

*       BL1_THETA= 0.D0

c= Elevation angle, i.e. the angle between beamline direction and its
c= projection onto (Z,Y) plane. If it equals to zero, the beamline remains
c= in horizontal plane, positive phi correspond to increasing X.

*       BL1_PHI  = 0.D0

c= Roll angle about the beamline direction vector, i.e. angle between line
c= formed by intersection of beamline (x,y)-plane with (Z,Y) plane and
c= beamline x axis. A positive angle forms a right hand screw with beamline
c= direction vector

*       BL1_PSI= 0.D0

*       x_start  = 0.d0
*       y_start  = 0.d0
*       z_start  = 0.d0

*       call buildbl( 'optics1',
*      &      x_start, y_start, z_start,
*      &      bl1_theta, bl1_phi, bl1_psi,
*      &      InitZone)

*       write(*,*)'REG1:INFO: use beam Ekin=',
*      &      BeamKineticE,
*      &      ' ;beam mass',
*      &      BeamMass

*       BeamKineticE = 100.0d0
*       BeamMass      = 0.93827231d0
```

```
*       call setebl( BeamKineticE, BeamMass, 5.6780d+03 )

*       call blmaxmat( nblnzmax )
*       if ( nblnzmax .gt. M_MAX ) then
*          write(*,*) 'beamline 1: ',
*     &          'nblnzmax as read from blmatmax ',
*     &          nblnzmax
*          write(*,*) 'is greater then M_MAX', M_MAX
*          stop 'STOP'
*       else if ( nblnzmax .lt. M_MAX ) then
*          write(*,*) 'beamline 1: ',
*     &          'nblnzmax as read from blmatmax ',
*     &          nblnzmax
*          write(*,*) 'is less then M_MAX', M_MAX
*       endif

*       if ( nblnzmax .gt. 0 ) then
*          call blmat(IMUN)
*       else
*          write(*,*) 'Number of zones for NuMI beamline <= 0'
*          stop
*       endif

*       call set_current_beamline(2)

*       bl2_theta = 0.0d0
*       bl2_phi   = 0.0d0
*       bl2_psi   = 0.0d0

*       x_start  = 0.d0
*       y_start  = -300.d0
*       z_start  = 3000.d0

*       InitZone = 2000

*       call buildbl( 'optics2',
*     &      x_start, y_start, z_start,
*     &      bl2_theta, bl2_phi, bl2_psi,
*     &      InitZone)

*       call setebl( BeamKineticE, BeamMass, 5.6780d+03 )

*       call blmaxmat( nblnzmax )
*       if ( nblnzmax .gt. M_MAX ) then
*          write(*,*) 'beamline 1: ',
*     &          'nblnzmax as read from blmatmax ',
*     &          nblnzmax
*          write(*,*) 'is greater then M_MAX', M_MAX
*          stop 'STOP'
*       else if ( nblnzmax .lt. M_MAX ) then
*          write(*,*) 'beamline 1: ',
*     &          'nblnzmax as read from blmatmax ',
*     &          nblnzmax
*          write(*,*) 'is less then M_MAX', M_MAX
*       endif

*       if ( nblnzmax .gt. 0 ) then
*          call blmat(IMUN)
```

```
*         else
*            write(*,*) 'Number of zones for NuMI beamline <= 0'
*            stop
*         endif

*         do i=1, st_nof_zones
*            call simple_tunnel_mat(i, IMUN(i+st_first_zone-1))
*         end do


c...  volume names
*         call set_current_beamline(1)
*         call blname(VOLNM(NFZPEX+1))

*         call set_current_beamline(2)
C*        call blname(VOLNM(NFZPEX+1))

      RETURN
      END


      SUBROUTINE BLGEOINIT(BEAMLINE_ID)
C.........................................................
C     MMBLB ELEMENT REGISTRATION
C     CALLED IF IND(13)=T
C
C-----
C     CREATED:     2001 BY KRIOL
C     MODIFIED:    17-OCT-2003 BY MAK
C     LAST CHANGE: 04-DEC-2003 BY NVM
C-----
C.........................................................
      IMPLICIT NONE
      INTEGER BEAMLINE_ID

C=== EXAMPLE OF REAL WORK (uncomment and modify * whereever needed) ===

*          INCLUDE 'DRIFT.INC'
*          INCLUDE 'RBENDEPB.INC'

*          EXTERNAL
*       &      DRIFT_NAME_FUNC,
*       &      DRIFT_INIT_FUNC,
*       &      DRIFT_MAT_FUNC,
*       &      DRIFT_FIELD_FUNC,
*       &      DRIFT_VOL_FUNC,
*       &      DRIFT_GEO_FUNC,
*       &      DRIFT_ZONENAME_FUNC

*          IF (BEAMLINE_ID .EQ. 1) THEN
*             CALL MARS_EL_REGISTER( 1,
*       &         DRIFT_NOF_ZONES,
*       &         DRIFT_NAME_FUNC,
*       &         DRIFT_INIT_FUNC,
*       &         DRIFT_GEO_FUNC,
*       &         DRIFT_MAT_FUNC,
*       &         NO_FIELD,
*       &         DRIFT_VOL_FUNC,
*       &         DRIFT_ZONENAME_FUNC )
```

```
*           CALL MARS_EL_REGISTER( 2,
*      &          RBENDEPB_NOF_ZONES,
*      &          RBENDEPB_NAME_FUNC,
*      &          RBENDEPB_INIT_FUNC,
*      &          RBENDEPB_GEO_FUNC,
*      &          RBENDEPB_MAT_FUNC,
*      &          RBENDEPB_FIELD_FUNC,
*      &          RBENDEPB_VOL_FUNC,
*      &          RBENDEPB_ZONENAME_FUNC )
*
*       ELSE IF(BEAMLINE_ID .EQ. 2) THEN
*           CALL MARS_EL_REGISTER( 1,
*      &          DRIFT_NOF_ZONES,
*      &          DRIFT_NAME_FUNC,
*      &          DRIFT_INIT_FUNC,
*      &          DRIFT_GEO_FUNC,
*      &          DRIFT_MAT_FUNC,
*      &          NO_FIELD,
*      &          DRIFT_VOL_FUNC,
*      &          DRIFT_ZONENAME_FUNC )
*
*           CALL MARS_EL_REGISTER( 2,
*      &          RBENDEPB_NOF_ZONES,
*      &          RBENDEPB_NAME_FUNC,
*      &          RBENDEPB_INIT_FUNC,
*      &          RBENDEPB_GEO_FUNC,
*      &          RBENDEPB_MAT_FUNC,
*      &          RBENDEPB_FIELD_FUNC,
*      &          RBENDEPB_VOL_FUNC,
*      &          RBENDEPB_ZONENAME_FUNC )
*       END IF

      RETURN
      END


      SUBROUTINE TUNNEL_GEO(XLOCMAD,YLOCMAD,ZLOCMAD,XGLMAD,YGLMAD,
     &      ZGLMAD,NZ,BLINDX,ELINDX,ELBEGINS,ELLENGTH,ANGLE,PREVANGLE,
     &      NEXTANGLE)

      IMPLICIT NONE

      INTEGER NZ, BLINDX, ELINDX
      DOUBLE PRECISION XLOCMAD,YLOCMAD,ZLOCMAD,XGLMAD,YGLMAD,ZGLMAD,
     &      ELBEGINS,ELLENGTH,ANGLE, PREVANGLE, NEXTANGLE

      RETURN
      END
```

5.17   User Subroutine Examples

5.17.1   Simple Model of Beam on a Target

Example 1. Calculate antiproton production with 9-cm long 1-cm diameter copper target irradiated with 120-GeV proton beam. Beam R.M.S spot size is σ_x=0.005 cm and σ_y=0.007 cm. In addition to forced antiproton production the user is interested in energy deposition calculation including knock-on electron and e^+e^--pair production by hadrons. To create a file of antiprotons generated on the target

in a given phase space, the user adds in MAIN:

```
OPEN (UNIT=9,FILE='PBAR.OUT',STATUS='UNKNOWN')
```

and a few statements in the LEAK routine (Section 5.9). The geometry and scoring is described in the *standard* mode. The MARS.INP file can look as:

```
Pbar Cu Target, sigx=0.005, 02-Nov-1994
INDX T 5=T 12=T
NEVT 100000
ENRG 120.
IPIB 1 2
BEAM 0.005 0.007
SMIN 0.001 3.
MATR 'CU'
ZSEC 9.   51=3
NLTR 5
RSEC 0.002 0.005 0.01 0.1 .5
STOP
```

The geometry is very simple here, completely adequate to the *standard* mode. Instead of the above bining and writing to a file PBAR.OUT, one can use HBOOK (see MAIN) for analyses of energy deposition and generated antiprotons. Then the solid target can be described just as:

```
ZSEC 9.
RSEC 0.5
```

5.17.2   Beam Dump

Intense 800-GeV proton beam hits a graphite dump followed by aluminum and steel absorbers, by 210-cm vacuum gap, by 5-cm polyethylene slab, and finally by 1 meter of a wet dirt. The user is interested in energy deposition calculations, in maximum amount of the output, including partial distributions, temperature rise and estimation of a residual dose rate, in intermediate result dumps. A compound material is defined in the MIXTUR routine (Section 5.2). The routine REG1 (Section 5.4) defines the dump as surrounded with a steel shield sitting at the axis of a hypothetical cylindrically symmetrical tunnel with 30-cm thick concrete walls. With these three user routines a *standard* MARS.INP file can look as (*Option 1*):

```
NSUR   11
RZTS   6.15   6.15   -200.   200.
       200.   200.   -450.   450. 0.
       250.   250.   -450.   450. 0.
       6.15   60.     450.   450. 0.
       60.    200.    450.   450. 81.
       60.    60.    -200.   200. 0.
       230.   230.   -450.   450. 0.
       300.   300.   -450.   450. 0.
       400.   400.   -450.   450. 0.
       450.   450.   -450.   450. 0.
       470.   470.   -450.   450. 0.
```

```
Tevatron C0 Dump, 02/28/95
INDX T T 8=T
NEVT 500000 5
IPIB 1 2
ENRG 800.
SMIN 0.01 8.
BEAM 0.0416 0.0944
VARS 0.  1.   300.  4.E13
NMAT 6
MATR 'C' 'AL' 'FE' 'CONC' 'CH2' 'MIXT'
26=2.1
NLNG 7
ZSEC 350.  425.  475.  490.  700.  705.  805.
51=14 5 10 1 3 2 5 101=1 2 3 2 0 4 5
NLTR 9
RSEC 0.03 0.1 0.3 1.  3.5 10.  30.  150.  180.
PLOT 5.  300.  450.  805.
STOP
```

Option 2. The geometry description is simpler and the output is more sophisticated if one uses HBOOK activated in MAIN. The last lines in MARS.INP can be re-written as:

```
NLNG 7
ZSEC 350.  425.  475.  490.  700.  705.  805.
101=1 2 3 2 0 4 0 5
NLTR 4
RSEC 10.  30.  150.  180.
```

5.17.3   Specifying Volume Histograms

Put something similar to the Surface example in here.

5.17.4   Specifying Surface Histograms

6   MCNP and FLUKA Modes

The MCNP mode, already described in the Input File Section, is strongly recommended for the correct description of neutron interactions in the MARS15 code at neutron energies below 14.5 MeV, instead of the default BNAB one. To use this option, you must have the MCNP4C code with corresponding libraries obtained from the RSICC center in the USA [90] or NEA Databank in Europe [91] and installed on your group's computer.

To install the MCNP4C code itself, usually just Fortran and C compilers are needed. However, for the procedure described below, two additional softwares are required: a GNU **make** from the Free Software Foundation [92], and a **fsplit** utility for splitting a fortran code into separate subroutines. Some Linux distributions do not include it, therefore, a free utility source code from the BSD is included in the installation package as described below. All the files required for compilation can be downloaded from the secure section of the MARS web page [93].

6.1   MCNP4C Installation

The installation procedure for the MCNP4C code itself is described in the installation guide which supplements the manual on this code. A user must run the **install** procedure and follow the on-screen instructions carefully. While configuring and installing the MCNP4C, choose the following options:

 Dynamic memory: Off with default **mdas** size

 Geometry Plotter: Off

 Tally Plotter: Off

 64-Bit Data: Off

 Multiprocessing: Off

After completion this procedure and successful running the built-in test problems, next step is the MCNP4C library compilation.

6.2   MCNP4C Library Compilation

Create a new directory and copy from the MCNP4C directory the following components required to create the library:

- mcnpf.id

- prpr

- patchf

Copy to the same directory the following files downloaded from the MARS web page:

- GNUmakefile

- source.patch

- fsplit.c

- lsignal.c

Check GNUmakefile for correct commands and compiler options. If there is no a **fsplit** utility on your system, create it by means of the following command

cc -o fsplit fsplit.c

Now run the GNUmakefile and wait for several minutes until the procedure is complete. Upon its successful completion, a MCNP library called, for example, **libmcnp4Csun.a** will be created. It must be moved to the directory .../restricted/mcnp4c/os/lib, where "os" is sun, linux, dec, aix, or irix depending on the OS of your computer, and "..." can look like /home/mokhov.

6.3 Data Libraries

A set of evaluated neutron, photon, and electron data libraries of the MCNP4C distribution package must be copied into the .../restricted/mcnp4c/data directory. The following line must be inserted at the top of the existing file **xsdir**:

datapath=.../restricted/mcnp4c/data

where "os" and "..." are as above.

6.4 Running MARS in MCNP Mode

The **xsdir** file or link to it must be present in the directory where user runs MARS in the MCNP mode. To create a MCNP4C-compatible executable in that directory, user says

make rmars-mcnp-fems-os

instead of a usual "make" for a default BNAB mode, where "os" is as above. A corresponding MARS.INP in that directory must include

1. INDX 5=T,

2. MCNP materials description for those presented on the **MATR** card.

For example:
NMAT 3
MATR 'YOKE' 'AIR' 'CONC'
STOP
∗MCNP START
m1 6000 -0.001 14000 -0.001 25055 -0.004 26000 -0.982 &
28000 -0.010 29000 -0.002 cond=1
m2 7014 0.78443 8016 0.21076 18000 4.671E-3 6000 1.39E-4 gas=1
m3 1001 -0.006 6000 -0.030 8016 -0.500 11023 -0.010 13027 -0.030 &

98

Table 6: Neutron energy group numbers `NG` and lower energy boundaries `EG` (MeV) in the group at $E \leq 14.5$ MeV in the 28-group representation. Used with `IND(14)=T` and `IND(17)=F`.

NG=	1	2	3	4	5	6	7
EG=	14.	10.5	6.5	4.0	2.5	1.4	0.8
NG=	8	9	10	11	12	13	14
EG=	0.4	0.2	0.1	4.65E-2	2.15E-2	1.E-2	4.65E-3
NG=	15	16	17	18	19	20	21
EG=	2.15E-3	1.E-3	4.65E-4	2.15E-4	1.E-4	4.65E-5	2.15E-5
NG=	22	23	24	25	26	27	28
EG=	1.E-5	4.65E-6	2.15E-6	1.E-6	4.65E-7	2.15E-7	2.15E-9

14000 -0.200 19000 -0.010 20000 -0.200 26000 -0.014
∗MCNP END

For further details on materials description see the MCNP manuals.

Option A:

1. Put 14=T 17=F in MARS.INP.

2. Type `make` to link and create executable `rmars-bnab`.

3. Run it.

Option B:

1. Put 14=F 17=T in MARS.INP.

2. Put material description required by MCNP after the `STOP` card between `*MCNP START` and `*MCNP END` cards e. g.:
   ```
   *MCNP START
   m1 26000 1.0
   m2 14000 0.33 8016 0.62 1001 0.05
   *MCNP END
   ```
   Isotopes in MCNP are described in terms of ZZZAAA notation. First three digits give the isotope Z, while the last three give the atomic mass. Thus, 92238 describes uranium-238 isotope, 5010 describes boron-10 isotope, etc. If the last three digits are equal to zero, then the natural mixture of isotopes is assumed. Materials in MCNP are represented as mixtures. For example, line
   ```
   m1 26000 0.80 6012 0.20
   ```
   describes the first material (in the MARS numbering scheme !) with 80% of atoms being natural iron (26000) and 20% of atoms being carbon-12 isotope. Second example is for the same material, but percentage in mixture is given by weight:
   ```
   m1 26000 -0.95 6012 -0.05.
   ```
   Material with `IM=0` is equivalent to vacuum. Add as many MCNP material description lines as needed numbered corresponding to the material input sequence in the main MARS input section. For example, for two material, e.g., FE and LI, put the following two lines:
   ```
   m1 26000 1.0
   ```

```
m2 3007 1.0
```
For more advanced examples in setting up materials look into the MCNP4C manual and Web page.

3. File xsdir must be present in the run directory. This file gives addresses for ENDF/B-VI neutron/photon cross-section libraries and other data files.

4. Type `make rmars-mcnp` to link and create executable `rmars-mcnp`.

5. Run it.

6. MCNP subsystem creates three buffer files after the run. First, mcnp.buf, is temporary file generated at the intialization stage. It is re-written during each run. But if you think you found a bug in neutron transport, please send this file to us with a complete bug report. Second file is outp. It contains info on how the MCNP subsystem parses the input file. It might be useful for checking the atomic and weight percentage of material mixtures specified in the MARS.INP in the MCNP section. It is also re-created in each run. Third file, runtpe, contains binary information required for a hot restart. One can safely delete this file.

6.5 FLUKA Geometry Interface

MARS15 can read in an input geometry description in the FLUKA format and perform full MARS modeling in such a setup. An example of a MARS model of the VMS collider detector based on a FLUKA input is shown in Fig. 21.
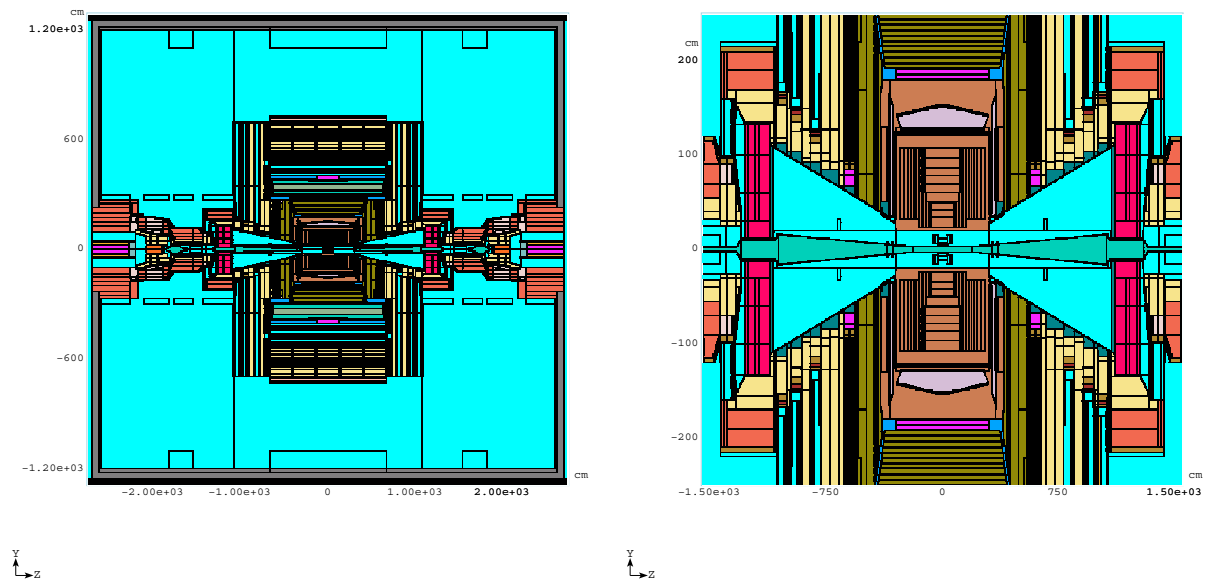

Figure 21: MARS model of the CMS detector based on a FLUKA input: overall view (left) and a central fragment (right).

7 LAQGSM and DPMJET Modes

7.1 LAQGSM

The 2003 version of the Los Alamos Quark-Gluon String Model code, LAQGSM03, was implemented into MARS15 for particle and heavy-ion projectiles at 0.01 GeV/A to 800 GeV/A [1]. Such processes as π^--capture and $\overline{p}$-annihilation on nuclei are treated down to essentially zero enrgy. Details of this event generators can be found elsewhere. This implementation provides the capability of full theoretically consistent modeling of exclusive distributions of secondary particles, spallation, fission and fragmentation products.

7.2 DPMJET

8 MAD-MARS Beam Line Builder

The MAD [94] lattice description language has become the *lingua franca* of computational accelerator physics. Any new developments in accelerator physics computational codes and libraries should have the requirements to read and understand lattice descriptions written in MAD. The ideas and modules of Ref. [95] are used in a new interface, which is able to read, parse, and store in memory MAD lattice descriptions, with the ability to generate an output file which translates those descriptions for input to MARS, and can also be used as input to other tracking and CAD applications.

The interface system—*MAD-MARS Beam Line Builder* (MMBLB)—reads a MAD lattice file and puts the elements in the same order into MARS geometry. Each element is assigned six functions which provide information about the element type/name, geometry, materials, field, volume and initialization. The user specifies the element type and an optional element name. If no name is specified, the element is considered to be a generic one. A building algorithm first tries to match the type/name pair and then substitute a generic element if needed. Once an element is described, it is registered with the system and its name is binded with the respective geometry, materials, volume and field descriptions. For each region search during tracking, MMBLB finds the corresponding type/name pair and calls its appropriate functions. MMBLB calculates a local rotation matrix $\mathcal{R}_i$ and a local translation vector $\mathcal{L}_i$. Then a global rotation matrix $\mathcal{M}_i$ and a position $\mathcal{P}_i$ are calculated and stored for each element

$$\mathcal{M}_i = \mathcal{M}_{i-1} \times \mathcal{R}_i, \quad \mathcal{M}_0 = \mathcal{U}$$
$$\mathcal{P}_i = \mathcal{M}_{i-1} \times \mathcal{L}_i + \mathcal{P}_{i-1}$$

where $\mathcal{U}$ is the unit matrix. $\mathcal{R}_i = \mathcal{U}$ for all elements, except RBEND and SBEND.

The newest version of MMBLB was substantially enhanced and is well described in the User's Guide [4]. The set of supported element types includes now almost all the elements supported by MAD. An arbitrary number of beam lines – arbitrary positioned and oriented – can be put in a MARS15 model. More sophisticated algorithms and new data structures enable more efficient searches through the beam line geometry. Tunnel geometry can now follow the beam line or be described independently of it.

For more details and use of MMBLB the user must read Ref. [4].

9 Multiprocessing

A parallel processing option has been developed and implemented into MARS15 [3]. It is based on the Message Passing Interface (MPI) libraries. Parallelization is job-based, i.e. the processes, replicating the same geometry of the setup studied, run independently with different initial seeds. A unique master process – also running event histories – collects intermediate results from an arbitrary number of slaves and calculates the final results when a required total number of events has been processed. Intermediate results are sent to the master on its request generated in accordance with a scheduling mechanism. The algorithm was tested on Unix and Linux clusters and demonstrated good performance.

10 Output of the Simulation *(to be Revised)*

There are MANY output files. Some are always produced. Others are produced only when certain options are selected via the input deck or user subroutines. Table xxx gives a list of all the possible output files, their default names, and their contents. This section goes through most of these files, giving a more detailed description of the contents of each one.

For example, histograms are only produced when requested via the main MARS.INP input deck; the MARS code is linked to the CERN library and uses the HBOOK package [96] to produce this output. The tabular format of a few of the output files can be easily cut-and-pasted into formats appropriate for various software plotting packages such as PAW, TOPDRAWER, GNUPLOT, XMGR, KALEIDAGRAPH and EXCEL. Other output files, such as those generated by use of the **TAPE** card in the input deck, have a less portable format which is fixed by non-user MARS routines which cannot be modified.

There are three levels of the MARS, version 13(95), output which are essentially self-explanatory:

- general output in the file MARS.OUT;

- a file MARS.HIST defined in MAIN and filled by HBOOK for further analyses with PAW and other tools;

- a few files to be used by graphics packages PAW and others (.GRA, .PLOT) and to be used in a consecutive run in a multi-stage case; the files are defined in the **??**

IND(1)=T – ALWAYS. The program calculates and prints distributions of e^+e^- and photon fluxes and of energy deposition density ϵ and related values: dose equivalent, instantaneous temperature rise ΔT at given initial temperature T_0=TEMPO and number of particles per beam N_0=AINT (see VARS), and contact dose due to induced radioactivity at N_0 beam intensity. Other values discussed in Section 10.1 are calculated independently of the IND(1) meaning.

10.1 The MARS.OUT File

A file MARS.OUT consists of two major sections: input status and printout of the Monte Carlo session results.

A. Input Status

The subroutine BEGINN prints the cards it has read and some calculated quantities to be used in the Monte Carlo session. Every value is provided with the corresponding keyword and title. The following is printed consecutively: code version number, date and time, title of the problem, options logical statement, requested numbers of incidents (events) and of intermediate dumps, incident particle and beam types, incident kinetic energy, particle fluence cutoff energies, star production threshold, incident particle type, initial coordinates and direction cosines of the beam spot center, R.M.S. beam spot sizes and angular spread, accuracy of boundary localization STEPEM and control parameter STEPH, point-like target efficiency, exponential conversion factor, initial temperature, beam intensity, number of hA generations to follow.

Then for each material the following values that have been either read in or calculated are printed: material index, material name, averaged atomic mass and atomic number, averaged electron density, ionization potential, critical energy and radiation length, contents of composite materials (if any), threshold energies for δ-electrons and direct e^+e^- pair production by charged hadrons, calculated

lengths for inelastic hadron-nuclear interactions at incident energy `E0` and ionization ranges at the cutoff energy `EM`.

If `IND(14)=T` information on low-energy neutrons is printed: content of the **NEUS** card, group cross-sections for each material, number of point-like detectors and their coordinates (if any).

If `IND(11)=T` there is a printout of the azimuthal grid and if `NOB`≥1 a printout of radial and longitudinal boundaries of macro-regions for particle spectra scoring.

If all of `RPLOT, ZPLO1, ZPLO2, ZPLO3` are not equal to zero, two lateral and one longitudinal cross-sectional views will be drawn.

B. Printout

First, for the $standard$ (r, z, ϕ) part of the considered geometry, a table is printed which indicates correspondence of region boundaries and index `N`, assigned to each region. Also printed are corresponding material and magnetic field indices. The calculational results are referenced to this table by a region number `N`.

The `MARS.OUT` file contains the results of the Monte Carlo session as a number of tables and single quantities. All results are normalized to one incident particle (to one event), only temperature rise and residual dose distributions are normalized to `AINT` incidents (events). The following tables are printed here consecutively:

1. Longitudinally integrated lateral distributions of charged and total star density and hadron flux, of partial and total energy deposition.

2. Laterally integrated longitudinal distributions of partial and total energy deposition, corresponding cumulative distribution and total energy deposited via the different channels: neutron reactions below EM, low-energy particles from the nuclear de-excitation processes, electromagnetic showers and ionization losses of charged hadrons and muons.

3. Three-dimensional star density distribution *(stars per cubic centimeter)* induced by charged hadrons and by all hadrons for momenta $\geq PSTAM$ with corresponding statistical errors (one R.M.S.). Collision estimator.

4. Three-dimensional total hadron flux distribution *(particles per centimeter squared)* for kinetic energy $\geq$`EM` with corresponding statistical errors.
   Track-length estimator.

5. Three-dimensional charged hadron flux distributions for kinetic energy above two thresholds, 0.0145 and 0.5 GeV as a default.

6. Three-dimensional e^+e^- flux distributions for kinetic energy above two thresholds, 0.001 and 0.005 GeV as a default, if `IND(1)=T`.

7. Three-dimensional photon flux distributions for kinetic energy above two thresholds, 0.001 and 0.005 GeV as a default, if `IND(1)=T`.

8. Three-dimensional muon flux distributions for kinetic energy above two thresholds, 0.005 and 0.5 GeV as a default, if `IND(10)=T`.

9. Three-dimensional neutron (E<0.0145 GeV) flux distribution *(particles per centimeter squared)* with corresponding statistical errors, if `IND(14)=T`.

10. Total number of stars produced in the system.

11. Laterally integrated longitudinal distributions of star density, of total hadron flux, and of charged hadron, e^+e^- and muon fluxes.

12. Three-dimensional distribution of energy deposition density brought by low-energy particles from de-excitation of nuclei, in *GeV/g*, if `IND(1)=T`.

13. Three-dimensional distribution of energy deposition density of electromagnetic showers produced by π^0 decays, by high energy $\delta-$rays and by prompt e^+e^- pairs from hadrons, in *GeV/g*, if `IND(1)=T`.

14. Three-dimensional distribution of energy deposition density from hadron and muon electromagnetic losses with limited energy transfer, in *GeV/g*, if `IND(1)=T`.

15. Three-dimensional distribution of energy deposition density due to neutron interactions below E<0.0145 GeV, in *GeV/g*, if `IND(14)=T`.

16. Three-dimensional distribution of total energy deposition density with corresponding statistical errors (one R.M.S.), in *GeV/g*, if `IND(1)=T`.

17. Three-dimensional distribution of dose equivalent, in *Rem*, if `IND(1)=T`.

18. Crude estimation of three-dimensional distribution of residual dose rate, in *Rad/hr*, after 30-day irradiation at the mean beam intensity `AINT` particles per sec and 1-day cooling. These are valid only for sufficiently thick systems, beyond a lateral thickness of $\geq \lambda_{in}$.

19. Three-dimensional distribution of instantaneous temperature rise at given initial temperature T_0=`TEMPO` and number of particles per a single beam pulse N_0=`AINT`, if `IND(1)=T`.

20. Longitudinal distribution of relative energy deposition by charged particles falling below the thresholds. Two distributions presented if `IND(1)=T`, are for the first smallest radial bin and for the rest of the system.

21. Three-dimensional distribution of the $relative$ statistical errors of star densities, fluxes and energy deposition densities.

22. Leakage data: number and energy of albedo hadrons, of punchthrough hadrons and of hadrons that escaped the sides of the system; leakage energy of low-energy neutrons and of electromagnetic showers; total leakage energy; energy balance.

23. Leakage energy spectra of different particles for the upstream plane, downstream plane and for the rest of the system.

24. If `NOB`$\geq$`1`, energy spectra of different particles in the pre-determined special regions.

25. Tables of particle spectra, of star density, of hadron, muon and low-energy fluence, of total energy deposition density, and of temperature rise, ready for use with graphics packages. The same tables are saved in separate `.GRA` files if activated in MAIN.

If `IND(8)=F`, the tables No. **5–7, 12–14, 17, 18, 20, 21** will be absent in the output.

If `IND(18)=T`, most of the above values will be printed in a compact form for the *extended* geometry regions as with `IND(8)=F`.

If the user defines geometrically complex insertions defined with REG1 routine as a supplement to the *standard* and *extended* geometries, additionally the program prints in a compact form most of the above quantities with corresponding statistical errors for all *non − standard* regions.

10.2 The MTUPLE Files

Say something here to describe the two MTuple output files.

Table 7: Default histogram ID at NOB=1.

Particle	Vertex	Fluence	Energy Dep.	Spectrum
n	1	101		301
h	2	102		302
Total h	3	103	203	
γ	4	104		304
e	5	105		305
Total EM	6	106	206	
μ	8	108	208	308

10.3 Histogram Output

In many cases it is worthwhile to initialize in the MAIN program the HBOOK package to use the output file MARS.HIST for interactive analysis with the PAW system. Then all the powerful features of that system as described in [97] can be used for comprehensive physics analysis of the run session. By default MARS sorts histograms by following classes:

The main program also allocates the dynamic memory for HBOOK and gives control to the whole system. Histogram set up and entry routines, defined in the file **m13hist.f**, can be re-defined by the user according to his/her specific needs.

10.3.1 Volume Histograms

- histogram type (vertex, fluence, energy deposition and energy spectrum);

- particle class (hadron, electromagnetic and muon);

- charge (neutrals, charged and total).

These histograms are filled for the whole system or for the NOB special regions defined in the MARS.INP file. The histogram list in the file MARS.HIST with their IDs and functional contents is obtained with the PAW command hi/list. The list is self-explainatory. Default histogram IDs are shown in Table 10.3.1. A few examples of corresponding analyses are given in Section 13.3.

ALL HISTOGRAMS DIVIDED BY VOLUME, SURFACE AREA, OR/AND DELTA-E.

HISTOGRAM TYPE:

IHTYP = 1 - VERTEX, (STAR/CM3) 1<ID<100

IHTYP = 2 - FLUENCE AT STEP, (1/CM2) 101<ID<200

IHTYP = 3 - ENERGY DEPOSITION, (GEV/G) 201<ID<300

IHTYP = 4 - ENERGY SPECTRUM, (1/CM2/DEL) 301<ID<400

IHTYP = 5 - SURFACE ENERGY SPECTRA, (1/CM2/DEL) 401<ID<500

IHTYP = 6 - SURFACE TIME SPECTRA, (1) 501<ID<600

PARTICLE INTERACTION CLASS:

ICL = 1 - HADRON

ICL = 2 - ELECTROMAGNETIC

ICL = 3 - MUON

CHARGE CONTROL:

NSG = 0 - NEUTRONS, PHOTONS

NSG = 1 - CHARGED PARTICLES

NSG = 2 - TOTAL

VOLUMETRIC:

HISTOGRAM ID = NU1+NSG+3*(ICL-1)+100*(IHTYP-1)+1000*(NRE-1)

10.3.2 Surface Histograms

SURFACE:

HISTOGRAM ID = NU1+NSG+3*(ICL-1)+100*(IHTYP-1)+1000*(NSUR-1)

NU1=1 (DEFAULT)

PARTICLE SPECTRA HISTOGRAMMING IN 80 BINS: 75 LOG + 5 LIN

NEUTRONS: 28 (1.E-11 - 0.0145) + 52 (0.0145 - E0) GEV

OTHERS: 80 (5.E-4 - E0) GEV

NHSPE=0 - dN/dE, DIVIDED BY DEL=DELTA(E) IN (1/CM2/GEV)

E*dN/dE, DIVIDED BY DEL=DELTA(LOG10(E))

FOR NEUTRONS IF IND(14)=T IN (1/CM2)

NHSPE=1 - E*dN/dE, DIVIDED BY DEL=DELTA(LOG10(E))

FOR ALL PARTICLES IN (1/CM2)

SURFACE CROSSING ESTIMATOR (IHTYP=5, 6):

SURFACE TIME SPECTRA IN (TMIN-TMAX) INTERVAL (sec)

HISTOGRAMS GENERATED ARE UNIFORM IN 80 BINS (in nsec!)

10.4 Event Generator Output

Section 5.1 explained the use of the MARS stand-alone event generator. Here the output produced by that running mode is described in more detail. The output files produced are TMA.OUT, TDNDX.OUT,

The TMA file always contains a table with the heading AVERAGE MULTIPLICITY AND EN-ERGY, with columns labeled by produced particle type, and several rows. The rows are defined as

```
        BLACK                   I haven't a clue
        GREY                    likewise
        SHOWER                  enter appropriate descriptions here
        FORW,CS>0
        X>0,BT>0.2
        X<0,BT0.2
        TOTAL
        E(X>0,BT>0.2)
        E(X<0,BT>0.2)
        ETOT
        SUBCUT NO.
        SUBCUT E
        ELAE,EBIND
        YIELD(pi+ ..)
        YIELD(pi- ..)
        SUM
        EG,EXGAM,EXWGA
```

If the user set IDNDX=2 in routine MARS1400, then the TMA file also contains a series of seven tables. Each of the seven tables is labeled by J=1, J=2, etc., where the value of the index identifies the produced particle type; these are the same seven listed at the top of the output file. Each table has the structure

```
 DEGR=0.0    5.00      10.00      15.00      20.00      30.00      60.00      90.00     135.00     180.00

 EE(GEV)                                               D2N/DEDO                                           qDN/DE
 5.000     7.49E-02   8.06E-02   9.04E-02   8.61E-02   1.77E-01   4.30E-01   3.31E-01   2.03E-01   9.10E-02   3.13E-01
 10.000    3.64E-02   2.47E-02   4.10E-03   0.00E+00   0.00E+00   0.00E+00   0.00E+00   0.00E+00   0.00E+00   1.30E-02
```

The tables hold the production cross section $d^2N/dEd\Omega$ for the given particle type, as used by the generator, with the columns being bins in degrees, and the rows being bins in energy. The last column is dN/dE. The number of rows which appears is determined by the value of NEVBIN set in routine MARS1400. If instead the user set IDNDX=3 in routine MARS1400, then the TMA file contains a different series of seven tables, where the columns are Ed^3N/d^3P in bins of P_t and the rows are bins in X_f. The last column is dN/dX

11 Graphical-User Interface

A Graphical-User Interface, MARS-GUI-SLICE, has been developed. It is based on *Tcl/Tk* and is linked in to the user's executable, however it is active only when specific flags are set in the input file. When the interface is active, no events are generated, but the user's encoded geometry is probed and visually displayed. The interface displays all the details of the encoded geometry, showing the encoded zone numbers, materials and magnetic fields; it is a valuable tool for checking complex geometries before executing event generation. During event generation runs, the user can specify output files holding histograms and particle tracks; these output files can be opened by the *GUI* interface, post-run, and projected onto the visual display of the geometry. The main MARS-GUI-SLICE features are:

- Two-dimensional geometry slice and magnetic field view on a graphical display panel (GDP). Maximum and minimum coordinates along each axis and maximum field components are provided for the given view in corresponding entry fields (EF). They are changed automatically by grabbing a desirable view box on the GDP holding a *CTRL* key and clicking with the mouse left button at the two diagonal box corners. Alternatively, the lower and upper view boundaries can be typed in the EF along with a binning of the magnetic field grid seen on the GDP. There is a *1:1 scale* check field (CF) to return to a *natural* scale.

- A slice plane is chosen by a corresponding radio-button. A magnetic field view can be interactively turned ON and OFF in a corresponding CF.

- Materials distribution in a given view is represented in a color or black and white *wire-frame (contour) mode* or in a *color region-filled mode*, with the mode chosen by a corresponding radio-button. By clicking a corresponding button, a *Materials* window is created, with the CF displaying material index and name and select boxes (SB) showing color of each material in the given view on the GDP. The pre-set materials colors can arbitrary be modified in the corresponding SB individually for each material. The colors can be reset individually or globally. Changing the view automatically adjusts the material info in this window.

- By clicking a left mouse button at any point of the GDP, a *Point Info* window is created with information display fields (IDF) containing coordinates, region number, material name and index, magnetic field module and a value of histogram (see below) for this point. This window keeps the position intact.

- Particle tracks in the given view can be displayed on the GDP by loading a *.PLOT file generated by MARS. Similar to materials, by clicking a corresponding button, a *Particles* window is created, with information similar to the *Materials* window: particle ID, name, color and SB displaying color of each particle and allowing color modification. A corresponding CFs allow turning ON and OFF any ID and global track visibility. One can examine tracks by clicking a middle mouse button at any track point on the GDP. A *Track Info* window is created with IDF containing the particle ID, name, as well as the current energy, statistical weight and coordinates at the point.

- After the run, a variety of calculated 2-D histograms can be loaded and overlaped with the geometry view on the GDP. A *.HBOOK file is loaded in the *Load Hist* window and a desirable histogram is selected there from the IDF list by its ID or name. The geometry/histogram view is now handled as a whole. The *Point Info* window allows now for a detailed examination of the histogram values even within the same decade (color).

- The view can be inverted both vertically and horizontally.

- One can add arbitrary texts all over the GDP with a *Text* window activated by the mouse right button. Variety of fonts can be chosen there. Fonts, subscripts and superscripts are handled as in the XMGR plotting tool [98]. Text can be modified, moved around the GDP or deleted.

- Up to 20 of the GDP views can be stored and restored back and then viewed by clicking $<<$ or $>>$ buttons.

- The GDP can be saved as an Encapsulated Postscript file by clicking the *Print* button. The entire window or its arbitrary fraction can be XV-grabbed by clicking the *Grab* button.

- A version exists for a 3-D solid body representation [99].

This requires to have Tcl/Tk installed. Tcl/Tk is scripting language and graphical user interface designer, developed by Dr. J. Ousterhout. It is free software and anyone can get copy from Scriptics web site [100]. At least version 8.2 is required. Six variables in GNUmakefile might require changes in order to accommodate local installation:

- TCL_VER - Tcl version. Version 8.2 is known to work and it is the default value.

- TCL_INC - location of the Tcl include files.

- TCL_LOC - location of the Tcl include files.

- TK_VER - Tk version. Version 8.2 is known to work and it is the default value.

- TK_INC - location of the Tk include files.

- TK_LOC - location of the Tk library files.

- TCLTK_LST - list of Tcl/Tk library files. As default it consists of tcl and tk library -ltk$(TK_VER) -ltcl$(TCL_VER). Remember to put Tk library in linkage script before Tcl one.

All interface is inside one Tk/Tcl script called mf.tcl, which should be stored into MARS data directory.

11.1 How to use it

In order to run just set second control flag **CTRL** to 1 in the MARS.INP and run MARS executable - interface window will show up. Remember to set DISPLAY environment variable properly if you're working on remote computer via network.

11.2 Interface features

Interface main features:

- Draw magnetic field and two-dimensional geometry slice. Activated by pressing *Draw* button or *d* or *D* key. The picture of the main window is shown below
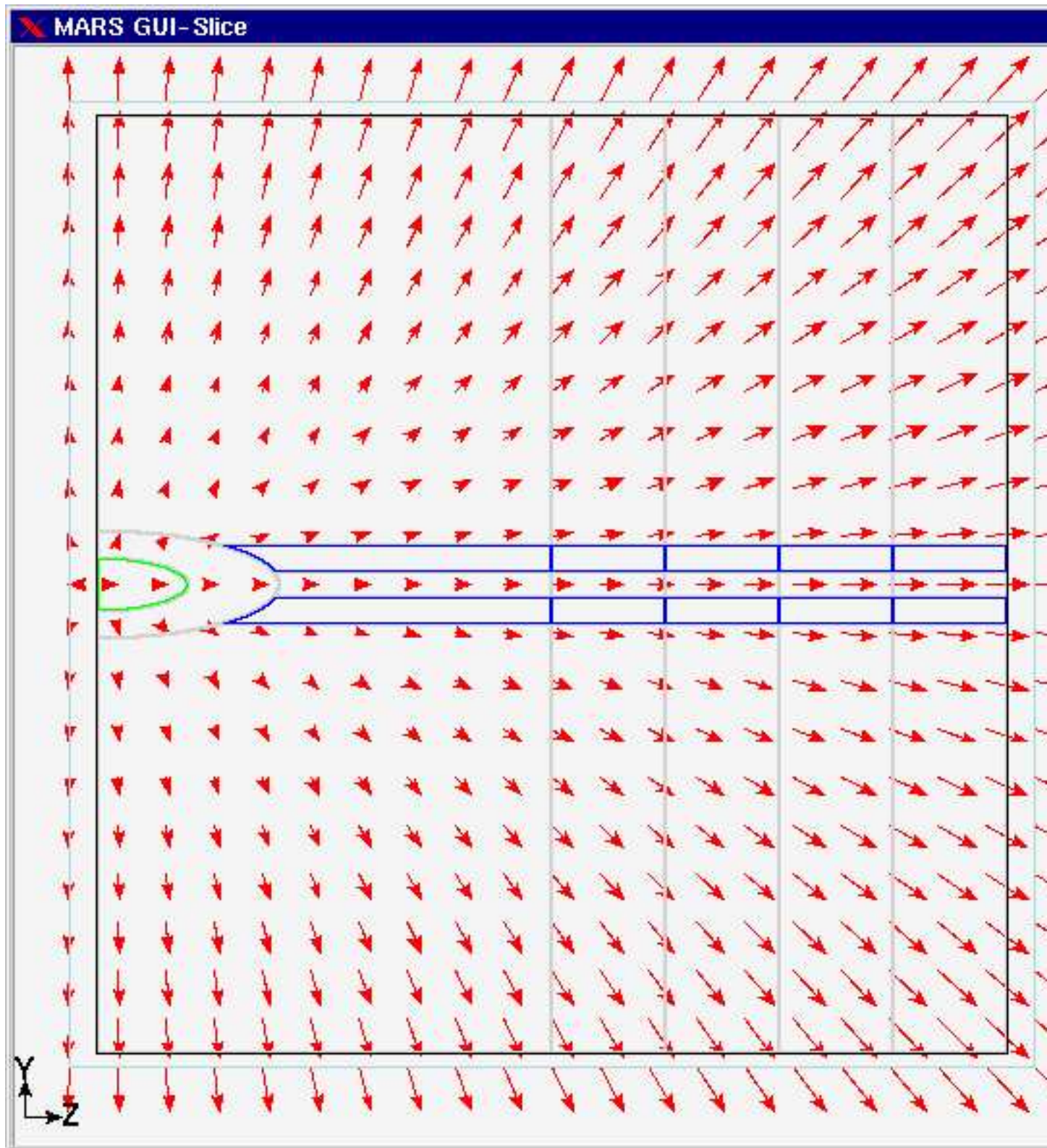
- The panel upper fields inform user about maximum field components and maximum field value

BH_max(T)	BV_max(T)	B_max(T)
20.600	36.050	41.521

- User can change the minimum and maximum values for each coordinate axis together with number of points used to draw magnetic field. It can be done using entries located on the right side of the interface. It is worth to mention also that

NBx	NBy	NBz
20	20	20
Xmin(cm)	Ymin(cm)	Zmin(cm)
−36.05	−36.05	−0.6
Xmax(cm)	Ymax(cm)	Zmax(cm)
36.05	36.05	20.6
◆ Y−Z	◇ X−Z	◇ X−Y

- User can set what axis slice will be activated. It can be done using radio buttons on the right side of the interface or pressing *x,X*, *y,Y* or *z,Z* keys respectively. Also the slice coordinate can be set using respective entry fields.

| ◆ Y−Z	◇ X−Z	◇ X−Y
X= 0.	Y= 0.	Z= 0.

- The magnetic field could be interactively turned ON and OFF by user using the button.Magnetic field is ON by default

| Magnetic field	ON

115

- Switch from so-called "Natural" to separately scaled view. In "Natural" mode program tries to preserve the same scale over horizontal and vertical axes. The horizontal axis counts as primary one, therefore when button "Natural" is pressed, vertical axis boundaries will be changed to accommodate changes in horizontal axis thus keeping the same overall scale. When "Natural" is turned OFF, you can set horizontal and vertical boundaries separately. Natural view is OFF by default.

Natural scale	OFF

- Checking coordinates, region number, material index and magnetic field module just clicking left mouse button on field/geometry map. It keeps the position intact and updates

Point info

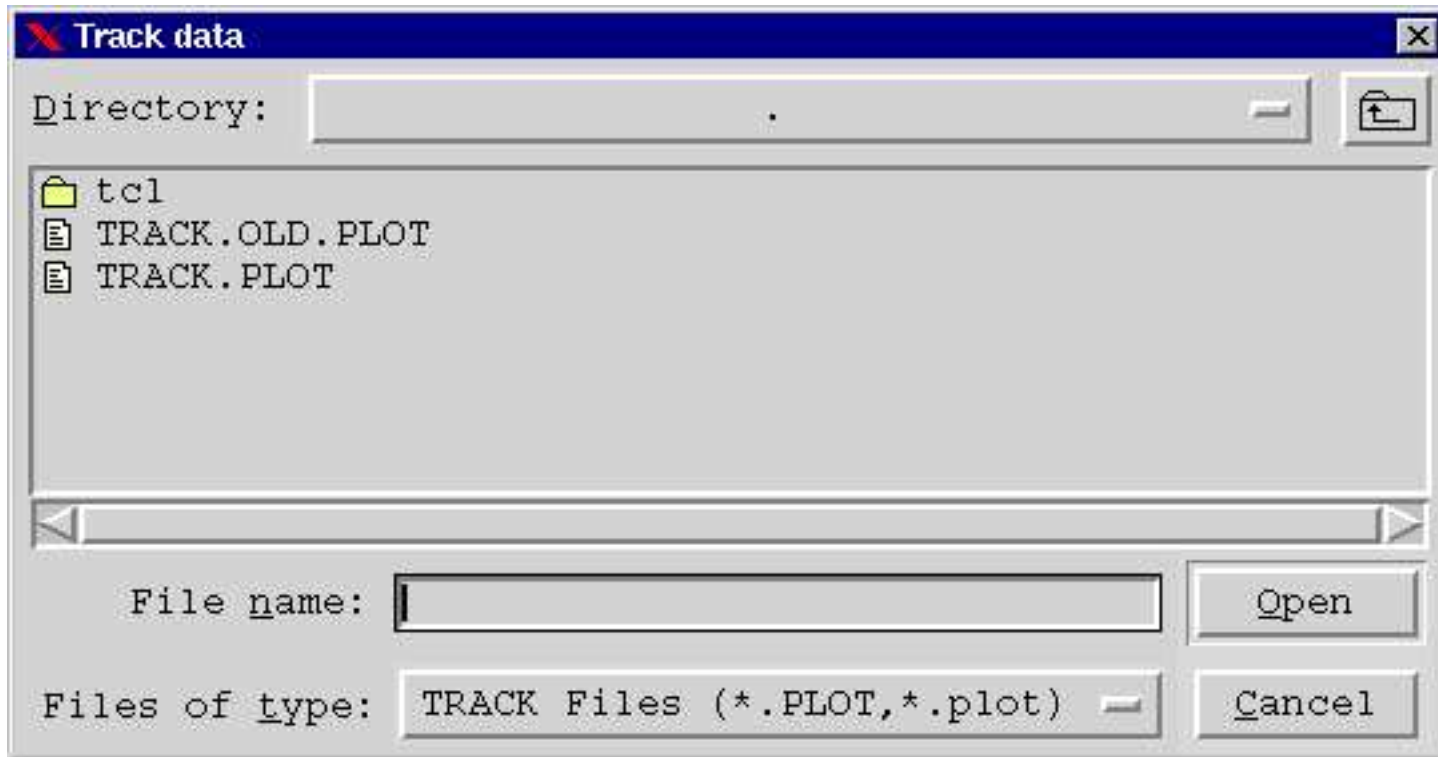|		Point Info		
NREG	IM	B(T)	Z(cm)	Y(cm)
11	0	23.227	8.813	21.486
Close				
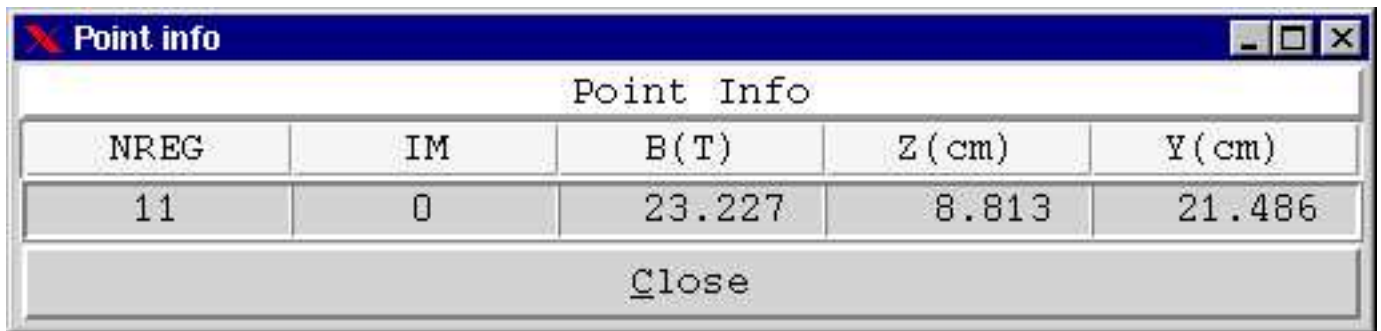
- Saving the drawing as well as ran external X grabber to create a picture as Encapsulated Postscript [101]. Saving the image as Encapsulated Postscript could be done by pressing *Print* button. User will get the dialog where he or she will be able to set EPS file name. You also can start external image grabber (XV by default) just pressing *Grab* button.
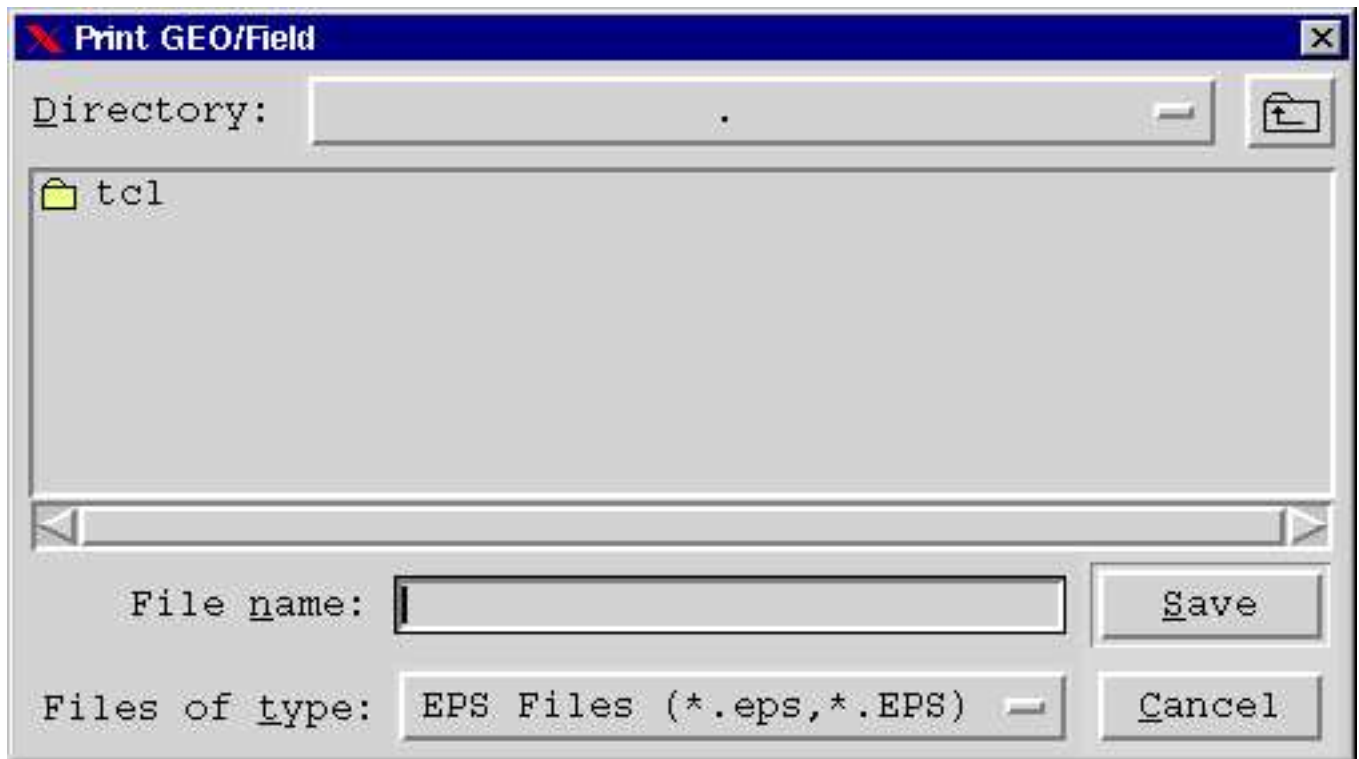
- User is able to load particle tracks as produced by MARS. Stored tracks file usually has **.PLOT** extension. Also there is possibility to turn on or off the tracks visibility. All of that can be done in main panel using "Load Track" button and switch.


The same as "Print" dialog will appear with prompt to enter track file name

Track data

Directory: .

```
📁 tcl
📄 TRACK.OLD.PLOT
📄 TRACK.PLOT
```

File name: |

Files of type: TRACK Files (*.PLOT,*.plot)

Open

Cancel

- There are two information panels can be started - one for materials and another for particles.

```
Materials        Particles
```

- Materials info panel allows you to examine which material is drawn in which color, turn ON and OFF interactively what material to show. It can be done by pressing the buttons on the panel left. The individual material color can be altered by pressing the colored square on the panel right.
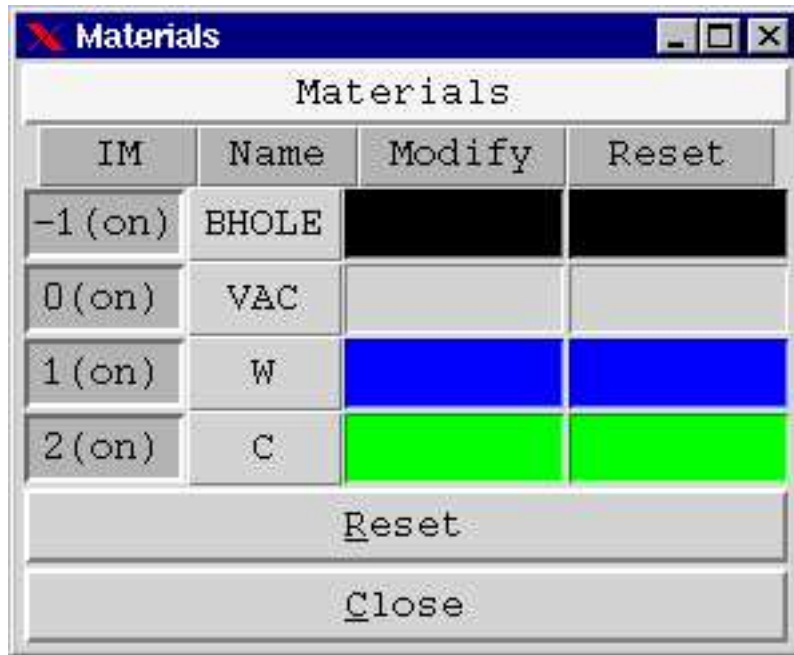
Of course, if you change the view and number of materials will be changed, the panel will adjust automatically

- Particles panel has the same properties and allows user to pick particular particles to show and change tracks color interactively

- User can examine tracks by pointing mouse and pressing middle button. panel with track information will appear and particle type, energy, weight and position will be shown.

- User can add texts all over the picture. One can do it by holding Shift key and pressing mouse right button simultaneously. Text will be drawn where the mouse pointer is positioned at this moment. Dialog window will appear where text can be entered


The same convention as in well known XMGR drawing tool [98] is used. Namely, \n will switch you to font number n, \S will switch text to superscript mode, \s will switch text to subscript mode and \N will return text back to normal position. Example above will produce the next text


- You also can choose fonts interactively by pressing Fonts button. Dialog will occur which will show you what fonts are available. Font numeration in the dialog is exactly the same used in text entry field
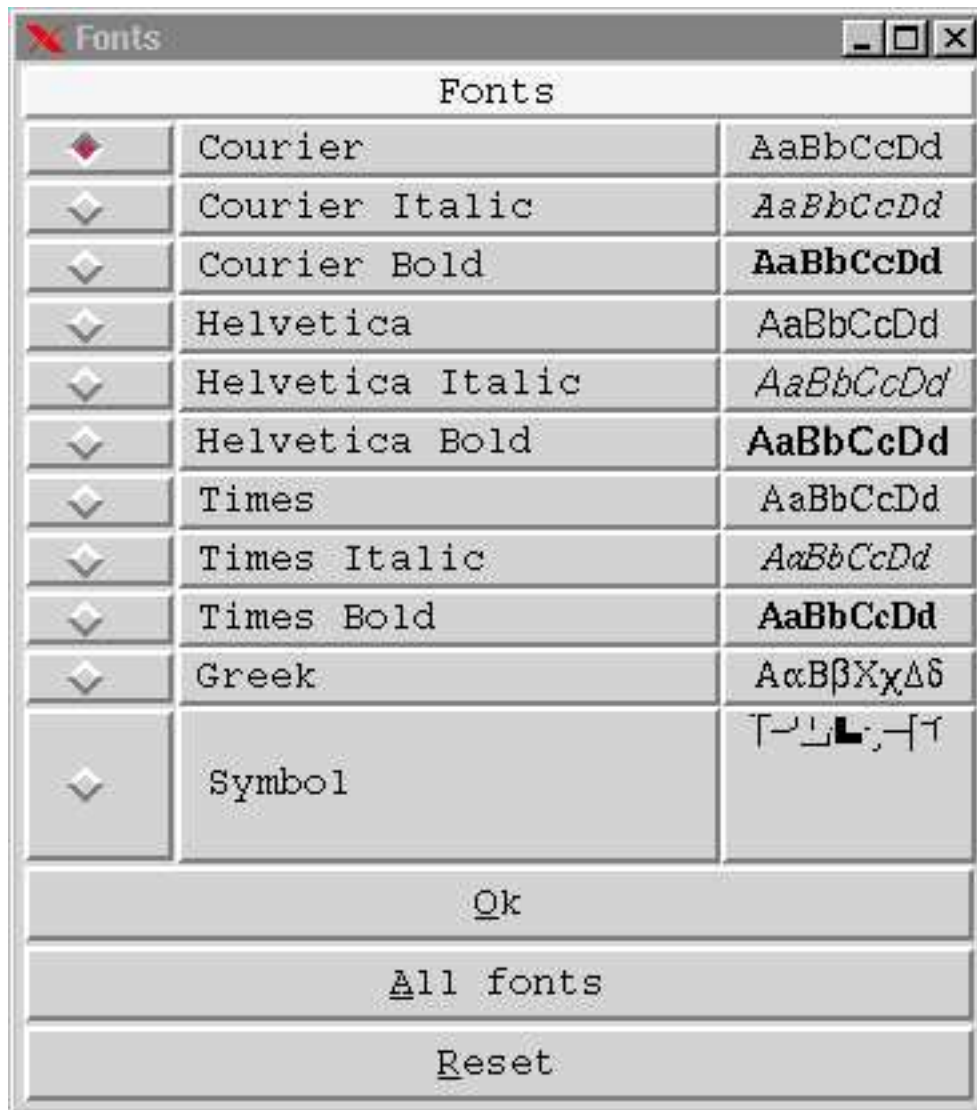
- Texts can be moved around the picture by grabbing them with third mouse button and dragging them around. User can delete texts by holding Ctrl key and pressing third mouse button on text fragment to be deleted.

- Ability to zoom in part of the picture using mouse. Just hold `Crtl` key while marking the opposite corner of zooming box and then redraw the geometry.

- Possibility to view the inverted image. You can use the following buttons to set either horizontal or vertical inversion and then redraw the geometry.
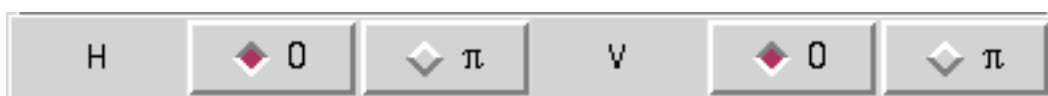
- The ability to draw the geometry with filled or transparence colors, or even in colorless (black&white) mode using next radiobuttons panel.
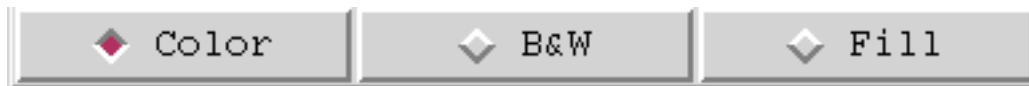
- The possibility to overlap the geometry view and histogram view. User could open histogram after MARS batch runs and draw the result of calculations on top of geometry

- Note, that it is user responsibility to make picture and histogram coherent. One can open .hbook file, check the histogram list and choose what histogram to load.
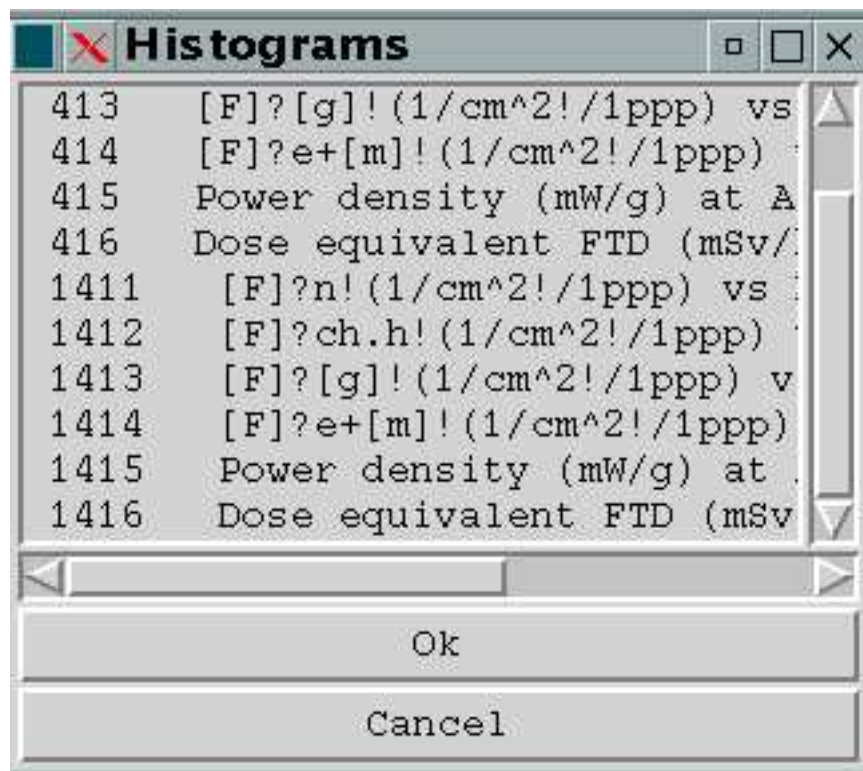
- The possibility to store and restore back number of previous geometry views and then move between them using one mouse click. That can be done using next buttons

<<   >>

12   Additional Code Topics

12.1   Calculating Volumes

The code calculates volumes in arbitrary (overlapped) regions in a short Monte Carlo session once one puts CTRL 3=1 in the MARS.INP file. As described in Section-4, these regions are contained in a cylinder defined by the the the **RZVL** card. The results go into an output file VOLMC.NON in the format ready for inclusion into the user routine VFAN. Statistical errors for the calculated volumes are also included to see if a number of events NVTRIAL in this session is enough.

Here is an example of a VOLMC.NON to be included in VFAN:

```
VOLUME(       7) =   0.8712684D+02   ! +/-   1.39%,   M=        1
VOLUME(       8) =   0.8915421D+02   ! +/-   1.37%,   M=        2
VOLUME(       9) =   0.8840023D+02   ! +/-   1.38%,   M=        3
VOLUME(      10) =   0.8766300D+02   ! +/-   1.38%,   M=        4
VOLUME(      11) =   0.8761274D+02   ! +/-   1.38%,   M=        5
VOLUME(      12) =   0.8754572D+02   ! +/-   1.38%,   M=        6
VOLUME(      13) =   0.8628908D+02   ! +/-   1.39%,   M=        7
VOLUME(      14) =   0.8806513D+02   ! +/-   1.38%,   M=        8
VOLUME(      15) =   0.8917097D+02   ! +/-   1.37%,   M=        9
VOLUME(      16) =   0.8922123D+02   ! +/-   1.37%,   M=       10
VOLUME(      17) =   0.8677498D+02   ! +/-   1.39%,   M=       11
VOLUME(      18) =   0.8863480D+02   ! +/-   1.37%,   M=       12
VOLUME(      19) =   0.8769651D+02   ! +/-   1.38%,   M=       13
VOLUME(      20) =   0.8774678D+02   ! +/-   1.38%,   M=       14
VOLUME(      21) =   0.8752896D+02   ! +/-   1.38%,   M=       15
VOLUME(      22) =   0.8853427D+02   ! +/-   1.38%,   M=       16
VOLUME(      23) =   0.9057840D+02   ! +/-   1.36%,   M=       17
VOLUME(      24) =   0.8818241D+02   ! +/-   1.38%,   M=       18
VOLUME(      25) =   0.8972389D+02   ! +/-   1.37%,   M=       19
VOLUME(      26) =   0.8778029D+02   ! +/-   1.38%,   M=       20
VOLUME(      27) =   0.4521213D+03   ! +/-   0.61%,   M=       21
VOLUME(      28) =   0.4606161D+03   ! +/-   0.60%,   M=       22
VOLUME(      29) =   0.4611858D+03   ! +/-   0.60%,   M=       23
VOLUME(      30) =   0.4583374D+03   ! +/-   0.60%,   M=       24
VOLUME(      31) =   0.4584715D+03   ! +/-   0.60%,   M=       25
VOLUME(      32) =   0.4570305D+03   ! +/-   0.61%,   M=       26
VOLUME(      33) =   0.4621576D+03   ! +/-   0.60%,   M=       27
VOLUME(      34) =   0.4569300D+03   ! +/-   0.61%,   M=       28
VOLUME(      35) =   0.4548021D+03   ! +/-   0.61%,   M=       29
VOLUME(      36) =   0.4573154D+03   ! +/-   0.61%,   M=       30
VOLUME(      37) =   0.4591919D+03   ! +/-   0.60%,   M=       31
VOLUME(      38) =   0.4571311D+03   ! +/-   0.61%,   M=       32
VOLUME(      39) =   0.4592087D+03   ! +/-   0.60%,   M=       33
VOLUME(      40) =   0.4649390D+03   ! +/-   0.60%,   M=       34
VOLUME(      41) =   0.4573824D+03   ! +/-   0.61%,   M=       35
VOLUME(      42) =   0.4620738D+03   ! +/-   0.60%,   M=       36
VOLUME(      43) =   0.4534784D+03   ! +/-   0.61%,   M=       37
VOLUME(      44) =   0.4571981D+03   ! +/-   0.61%,   M=       38
VOLUME(      45) =   0.4580358D+03   ! +/-   0.60%,   M=       39
VOLUME(      46) =   0.4558744D+03   ! +/-   0.61%,   M=       40
VOLUME(      47) =   0.1557392D+03   ! +/-   1.04%,   M=       41
VOLUME(      48) =   0.1562921D+03   ! +/-   1.04%,   M=       42
VOLUME(      49) =   0.1551360D+03   ! +/-   1.04%,   M=       43
VOLUME(      50) =   0.1569288D+03   ! +/-   1.03%,   M=       44
VOLUME(      51) =   0.1539464D+03   ! +/-   1.04%,   M=       45
```

```
VOLUME(    52) =   0.1565435D+03    ! +/-   1.03%,  M=       46
VOLUME(    53) =   0.1562586D+03    ! +/-   1.04%,  M=       47
VOLUME(    54) =   0.1573310D+03    ! +/-   1.03%,  M=       48
VOLUME(    55) =   0.1549015D+03    ! +/-   1.04%,  M=       49
VOLUME(    56) =   0.1559905D+03    ! +/-   1.04%,  M=       50
VOLUME(    57) =   0.5464025D+03    ! +/-   0.55%,  M=       51
VOLUME(    58) =   0.5460674D+03    ! +/-   0.55%,  M=       52
VOLUME(    59) =   0.5474246D+03    ! +/-   0.55%,  M=       53
VOLUME(    60) =   0.5488823D+03    ! +/-   0.55%,  M=       54
```

12.2   Supplementary Routines

Section on supplementary routines available from the MARS libraries: ATNM, RM48, RNDM, POISSN, NORRAN, NORMCO, FINEXT, EINT, FINT2D, SGINT1, ISOTR,

12.3   Interfaces to Other Programs

12.3.1   HBOOK

12.3.2   STRUCT

12.3.3   ANSYS

12.4   External Packages Required

13 Suggestions and Examples

13.1 Rules-Of-Thumb

It is obvious that the quality of the typical output of any code, for given physics model, depends on how the user built the calculational model and how he/she handled the code. The geometry description is of primary importance. Then, as stated above, the most essential parameters to control the calculational accuracy for given physics model are the number of incidents (primary events) NEVT and the inclusive/exclusive switches. Some other parameters also affect the computing efficency: the accuracy of boundary localization in iterative transport algorithm STEPEM, IND(6) option, cutoff energies, geometry details, histogramming etc. Naturally, the higher NEVT, the better the result will be. But here we come to a conradiction with both CPU time t_1 and time t_2 alotted for the whole problem. The strategy would be to keep both t_1 and t_2 as small as possible. There are a few rules for a user to get the best from the MARS code. These rules are:

1. Required NEVT is determined by a statistical error in a phase-space or geometrical region of interest. The calculated results in a given region N are statistically valid only if a R.M.S. statistical error $\delta \leq$ 20%. So, run until this condition is satisfied. Do short runs first to estimate required NEVT and play with DUMP output.

2. Before a long run, try to understand what a combination of inclusive/exclusive options provides highest computing efficiency.

3. Use as few as possible geometrical regions described in all the $standard$, $extended$ and $user-supplied$ sectors: CPU time grows almost linearly with the number of regions in the direction of predominant propagation of the particles.

4. Use as little as possible volume detector histogramming: it is rather time-consuming. At least minimize the number of bins.

5. Keep STEPEM$\simeq$0.1$\times t_{min}$, where t_{min} is a smallest linear size of the smallest region in the considered group of zones.

6. Use cutoff energies for each particle class as high as possible (to not damage result, of course), especially in bulk regions far from your regions of interest.

7. Use IND(15)=T for thick shielding calculations.

8. Use IND(6)=T and DLEXP$\neq$1 options for thick shielding calculations. Be carefull as with any biasing techiques, do short tests first.

9. Use IND(1)=F in a routine run to reduce amount of the output.

In some of these rules, the code takes care of those components in some effective manner anyway, but the user can reduce the CPU time drastically if he/she turns off the corresponding options.

13.2 Biasing and Other Control of Physics Processes *10/01*

This is a new section, intended to describe in detail how to control or modify the physics processes applied to various particle types as they are transported; for example what thresholds or other input deck settings modify which physics processes.

13.3 Examples

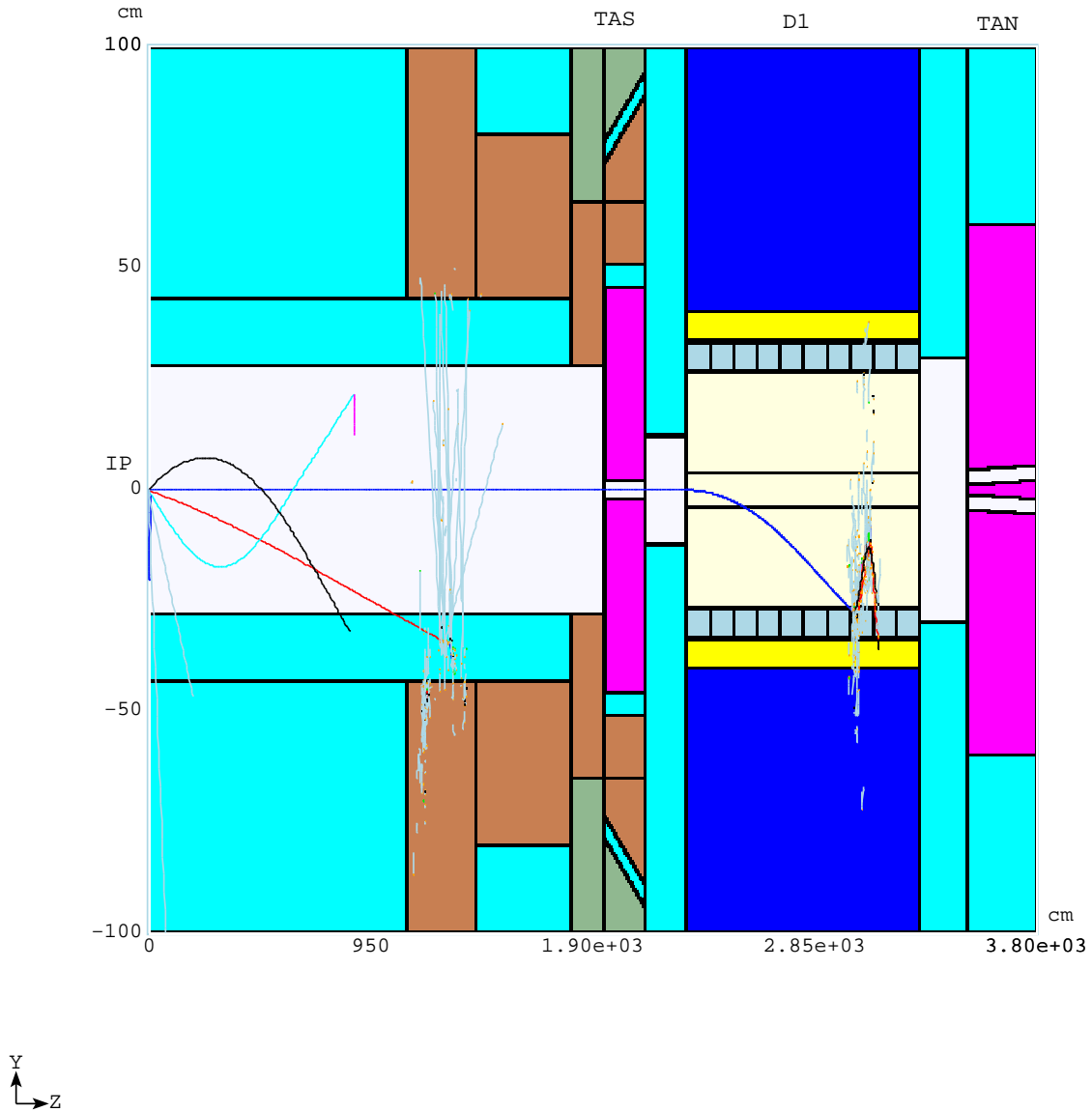## 13.4 2004

Shown in Fig. 22 and 23.

Figure 22: Particle tracks in the MARS15 model of the LHC-2 interaction region.

13.5 Very Old - to be updated

Results of MARS.OUT can be used as a final product. In addition, the created files .GRA and .PLOT are ready for acceptance by the popular graphics packages PAW, TOPDRAWER, GNUPLOT, XMGR and
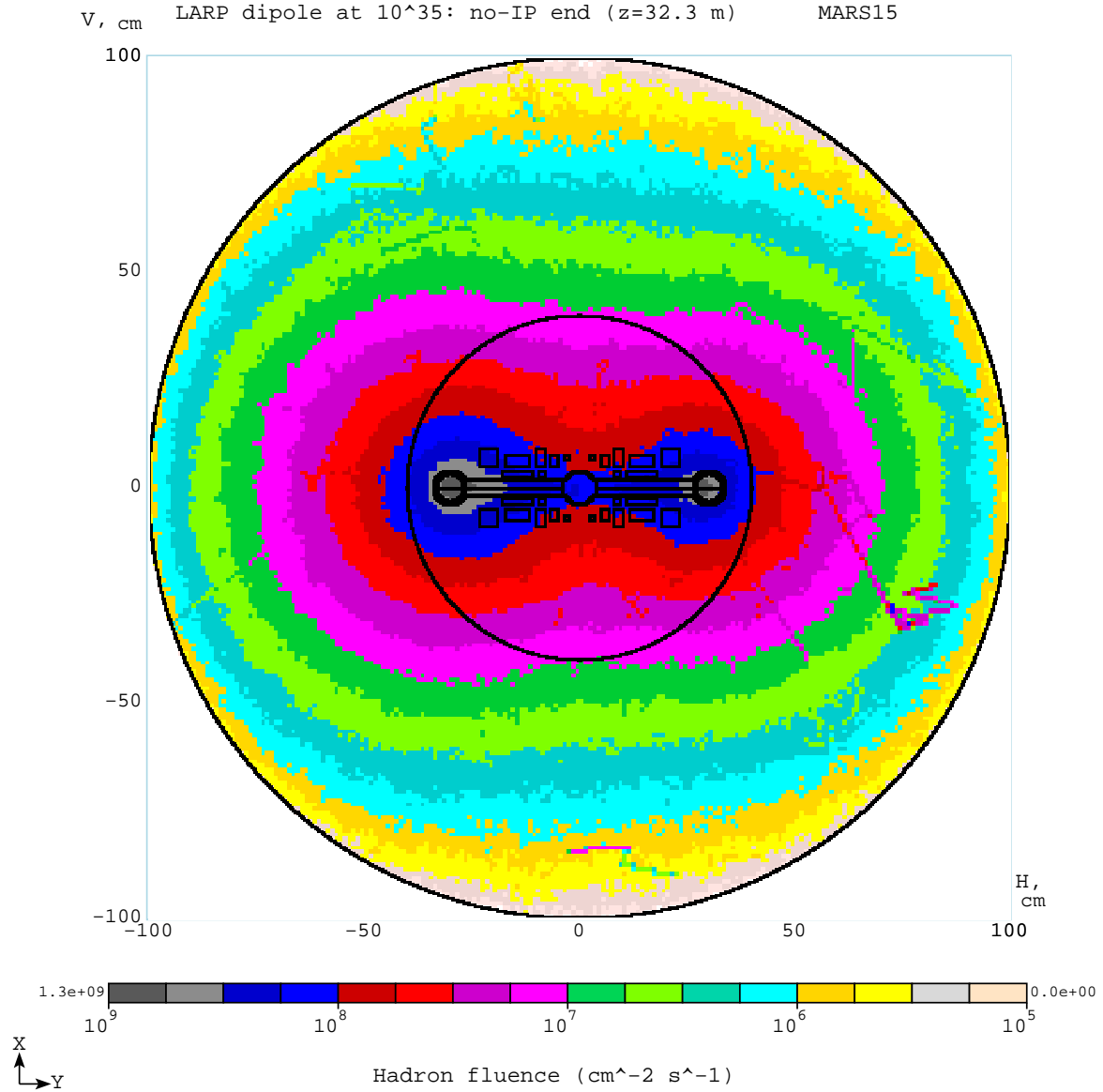
Figure 23: Hadron flux isocontours at E>1 MeV in the dipole-first model of the LHC-2 interaction region.

KALEIDAGRAPH to create high resolution plots for most of the distributions listed in the previous section. Figure 1 is a typical plot obtained via such an interface. The figure shows energy deposition rate in the C-layer of the SAMUS/WAMUS muon spectrometer of the D0 detector at Fermilab for two shielding configurations. Results are obtained with 4000 DTUJET93 events (0.9×0.9 TeV $p\tilde{p}$ collisions).

Figure 2 (histogram ID=203) represents energy deposition density in the forward region of the D0 detector at Fermilab. Interactions of particles, produced in the $p\tilde{p}$ collisions, with detector and accelerator components in this region (4 to 10 meters from the collision point) are the major source of backgrounds in the forward muon spectrometer. All the details of geometry and magnetic field were taken into account in MARS coupled here with DTUJET93. The figure highlights the hottest objects

and immediately indicates if the levels in the chamber are above the tolerable one. A similar plot for particle fluxes identifies the channels for radiation to reach critical regions and helps to find an appropriate absorber to plug them.

Recently a first-pass study [68] showed that the electromagnetic component of the backgrounds from $\mu{\rightarrow}e\nu\tilde{\nu}$ decays has the potential of killing the very attractive concept of a high-energy high-luminosity $\mu^+\mu^-$ collider unless there is significant suppression via various shielding and collimators in the detector vicinity. All simulations in the whole 80-m long inner triplet and in the detector were done with MARS, version 13(95). The simplified detector geometry used in the calculations is shown in Fig. 3.

In this application, MARS, allows an elegant way to handle the following processes:

- forced $\mu{\rightarrow}e\nu\tilde{\nu}$ decays in the beam pipe;

- tracking of created electrons in the beam pipe under influence of the magnetic field with emission of synchrotron photons along the track;

- simulation of electromagnetic showers in the triplet and detector components induced by electrons and synchrotron photons hitting the beam pipe;

- simulation of muon interactions (bremsstrahlung, direct e^+e^- pair production, ionization, deep inelastic nuclear interactions) and decays along the tracks in the lattice and detector;

- simulation of electromagnetic showers in the triplet and detector components created at the above muon interaction vertices;

- histogramming and analysis of particle energy spectra, fluences and energy deposition in various detector regions as well as in the whole IR.

Figure 4 shows calculated e^+e^- energy spectrum in the accelerator components in the vicinity of the detector. The huge peak sitting around 1 TeV represents the $\mu{\rightarrow}e\nu\tilde{\nu}$ decay spectrum with a tail at lower energies enriched by electrons and positrons of electromagnetic showers induced in the beam pipe and superconducting coils. Photons emitted due to synchrotron radiation along e^+e^- tracks in a strong 8 T magnetic field have an average energy around 1 GeV. The number of photons is about 300 times that for electrons and positrons.

References for numerous comparisons of MARS code predictions with data and with other codes are given in Section 2. A few rather interesting recent comparisons are presented below. Figure 5 shows attenuation of a dose rate along the 30-inch waveguide conduit adjacent to a Linac tunnel where an accidental beam loss (70 MeV, 6×10^{13} protons per sec) takes place. The results obtained with MARS and LAHET [102] are in a pretty good agreement. It is worthwhile to notice that the LAHET code has one of the best physics model for the intermediate energy range (20 to 600 MeV). An arsenal of non-analog techniques and a very detailed geometry description are necessary in these calculations making the agreement even more remarkable.

Another example is a recent intercomparison of four popular codes for high energy hadronic cascades studies: FLUKA [19], LAHET [102], GCALOR (mariage of GEANT and CALOR codes) [103] and MARS. A simple configuration was chosen: 100-GeV proton beam irradiating a 3-m long iron cylinder ($\rho = 7.4 g/cm^3$) with a radius of 50 cm followed by 200 g/cm^2 of ordinary concrete ($\rho = 2.35 g/cm^3$). Table refcomparetable gives the calculated total number of neutrons leaked through the front, side and

Table 8: Neutron leakage out of iron cylinder with a concrete shell.

Code	Front	Side	Back
MARS	153	4.2	0.96
FLUKA	118	3.6	1.02
LAHET	176	5.3	0.67
GCALOR	163	5.7	0.97

back surfaces. Corresponding leakage neutron spectra calculated with FLUKA using its most sophisticated physics model and with MARS are also in a rather good agreement (Fig. 6).

Comparisons for cascades induced by 20-TeV proton beam loss along the superconducting magnets in the SSC tunnel are shown in Fig. 7. Neutron spectra in the tunnel cross-section, obtained with the FLUKA and MARS codes, agree very well in the energy range spanning 10 decades. MARS calculations are also in a good agreement with GCALOR and FLUKA92 predictions in designs of shield configurations to reduce the background rates in the GEM detector at the SSC [104] and in the CMS muon system for the LHC project [105].

It is worthwhile to notice a recent remarkable achievement with the D0 detector at Fermilab [86]. A system to suppress backgrounds in the forward muon spectrometer, designed with the help of MARS coupled with the STRUCT code, has been installed in the vicinity of the experimental hall and has provided substantial reduction of the accelerator related particle fluxes in the detector, in an excellent agreement with MARS predictions.

14 Ongoing Developments

MARS is under active development, with new versions released every year or so. Registered users will be sent email announcing new releases, with a list of the features which have been implemented.

Some features now under development are

- XXXXXXXXXXX
- XXXXXXXXXXX
- XXXXXXXXXXX

14.1 Benchmarking

14.2 MARS Web Page and World-Wide Support

15 Acknowledgements

We express our gratitude to K.K. Gudima, M.A. Kostin, O.E. Krivosheev, S.G. Mashnik, I.L. Rakhno, A.J. Sierk, and S.I. Striganov for their contribution to this version of the MARS code.

References

[1] N. V. Mokhov et al., Recent enhancements to the mars15 code, Technical Report Fermilab-Conf-04/053, 2004.

[2] N. V. Mokhov, K. K. Gudima, S. G. Mashnik, I. L. Rakhno, and S. I. Striganov, Towards a heavy-ion transport capability in the mars15 code, Technical Report Fermilab-Conf-04/052, 2004.

[3] M. A. Kostin and N. V. Mokhov, Parallelizing the mars15 code with mpi for shielding applications, Technical Report Fermilab-Conf-04/054, 2004.

[4] M. A. Kostin, O. E. Krivosheev, N. V. Mokhov, and I. S. Tropin, An improved mad-mars beam line builder: User's guide, Technical Report Fermilab-FN-738, 2003.

[5] S. I. Striganov, On the theory and simulation of multiple coulomb scattering of heavy charged particles, Technical Report Fermilab-Conf-04/056, 2004.

[6] N. Nakao, N. Mokhov, K. Yamamoto, Y. Irie, and A. Drozhdin, Mars14 shielding calculations for the j-parc 3 gev rcs, Technical Report KEK-2004-1, 2004.

[7] M. A. Kostin and N. V. Mokhov, Non-neutrino particle flux in the minerva detector, Technical Report Fermilab-FN-745, 2004.

[8] N. V. Mokhov, Status of mars code, Technical Report Fermilab-Conf-03/053, 2003.

[9] N. V. Mokhov, Nauka, in *Proc. IV All-Union Conference on Charged Particle Accelerators*, volume 2, page 222, Moscow, 1975.

[10] N. V. Mokhov and V. V. Phrolov, Sov. J. Atomic Energy **38**, 226 (1975).

[11] R. P. Feynman, Phys. Rev. Lett. **23**, 1415 (1969).

[12] A. V. Ginneken, Weighted monte carlo calculations in thick targets, Technical Report FN-250, Fermilab, 1972.

[13] A. V. Ginneken, Casim – program to simulate transport of hadronic cascades in bulk matter, Technical Report FN-272, Fermilab, 1975.

[14] A. V. Ginneken, Calculation of the average properties of hadronic cascades at high energies (casim), in *Computer Techniques in Radiation Transport and Dosimetry*, edited by W. R. Nelson and T. M. Jenkins, 1978.

[15] N. V. Mokhov, Sov. J. Part. Nucl. **18**, 408 (1987).

[16] N. V. Mokhov, Inclusive simulation of hadronic and electromagnetic cascades in the ssc components, Technical Report SSC-SR-1033, SSC Central Design Group, 1988.

[17] A. N. Kalinovsky, N. V. Mokhov, and Y. P. Nikitin, *Passage of High-Energy Particles through Matter*, AIP, New York, 1989.

[18] W. R. Nelson, H. Hirayama, and D. Rogers, The egs4 code system, Technical Report SLAC-265, Stanford Linear Accelerator, 1985.

[19] A. Fasso, A. Ferrari, J. Ranft, and P. Sala, Fluka92, in *Proceedings of the SARE Workshop, Santa Fe, New Mexico, January 1993*, volume A349, Nucl. Instruments and Methods, 1994.

[20] N. V. Mokhov, S. I. Striganov, and A. V. Uzunian, Technical Report 87-59, IHEP, 1987.

[21] I. L. Azhgirey, N. V. Mokhov, and S. I. Striganov, Antiproton production for tevatron, Technical Report TM-1730, Fermilab, 1991.

[22] N. V. Mokhov, The mars12 code system, in *Proceedings of the SARE Workshop*, Santa Fe, New Mexico, 1993.

[23] S. L. Kuchinin, N. V. Mokhov, and Y. N. Rastsvetalov, Technical Report 75-74, IHEP, Serpukhov, 1975.

[24] I. S. Baishev, S. L. Kuchinin, and N. V. Mokhov, Technical Report 78-2, IHEP, Serpukhov, 1977.

[25] M. A. Maslov and N. V. Mokhov, Particle Accelerators **11**, 91 (1980).

[26] N. V. Mokhov, Energy deposition in targets and beam dumps at 0.1-5 tev proton energy, Technical Report FN-328, Fermilab, 1980.

[27] N. V. Mokhov, Technical Report 82-168, IHEP, Serpukhov, 1982.

[28] N. V. Mokhov, The mars10 code system: Inclusive simulation of hadronic and electromagnetic cascades and muon transport, Technical Report FN-509, Fermilab, 1989.

[29] N. V. Mokhov and J. D. Cossairt, Nucl. Instruments and Methods **A244**, 349 (1986).

[30] I. S. Baishev, I. A. Kurochkin, and N. V. Mokhov, Technical Report 91-118, IHEP, Protvino, 1991.

[31] I. L. Azhgirey, I. A. Kurochkin, M. A. Maslov, V. V. Talanov, and A. V. Uzunian, Technical Report 93-19, IHEP, Protvino, 1993.

[32] D. C. Wilson, C. A. Wingate, J. C. Goldstein, R. P. Godwin, and N. V. Mokhov, Hydrodynamic calculations of 20-tev beam interactions with the ssc beam dump, in *Proceedings of the 1993 Particle Accelerator Conference, IEEE*, pages 3090–3092, 1993.

[33] J. Ftacnik and M. Popovic, Parallel version of mars10 and mars12 codes, in *The 1994 April APS Meeting*, 1994.

[34] N. V. Mokhov, The mars code system user's guide, version 13(95), Technical Report FN-628, Fermilab, 1995.

[35] I. Baishev, A. Drozhdin, and N. Mokhov, Struct program user's reference manual, Technical Report SSCL-MAN-0034, SSC Laboratory, 1994.

[36] O. E. Krivosheev and N. V. Mokhov, A new mars and its applications, 1998.

[37] N. V. Mokhov et al., Mars code developments, 1998.

[38] N. V. Mokhov, Mars code developments, benchmarking and applications, in *Proc. of ICRS-9 International Conference on Radiation Shielding, October 17-22, 1999*, volume 1, pages 167–171, Tsukuba, Ibaraki, Japan, 2000, J. Nucl. Sci. Tech.

[39] N. V. Mokhov and A. V. Ginneken, in *Proc. of ICRS-9 International Conference on Radiation Shielding, October 17-22, 1999*, volume 1, pages 172–179, Tsukuba, Ibaraki, Japan, 2000, J. Nucl. Sci. Tech.

[40] N. V. Mokhov and O. E. Krivosheev, Mars code status, 2000.

[41] O. E. Krivosheev and N. V. Mokhov, Status of mars electromagnetic physics, 2000.

[42] N. V. Mokhov, S. I. Striganov, and A. V. Ginneken, Muons and neutrinos at high-energy accelerators, 2000.

[43]

[44]

[45]

[46]

[47] V. S. Barashenkov, Particle and nucleus cross-sections, Technical report, Dubna, 1993.

[48]

[49]

[50]

[51] N. V. Mokhov and S. I. Striganov, Beam dynamics and technology issues for $\mu^+\mu^-$ colliders, in *AIP Conference Proceedings*, page 234, Montauk, NY, 1995.

[52]

[53]

[54]

[55] N. V. Mokhov and S. I. Striganov, Model for pion production in proton-nucleus interactions, 1998.

[56]

[57] N. V. Mokhov and A. V. Ginneken, to be published.

[58]

[59]

[60]

[61] J. Ranft, Phys. Rev. **D51** (1995).

[62] J. Ranft, Gran sasso report, Technical Report AE-97/45, INFN, 1997.

[63] N. V. Mokhov, Muons at lhc. part 1: Beam dumps, Technical Report TIS–RP/TM/95–27, CERN, 1995.

[64] Y. Tsai, Rev. Mod. Phys. **46**, 815 (1995).

[65] N. V. Mokhov, G. I. Semenova, and A. V. Uzunian, Nucl. Instruments and Methods **180**, 469 (1981).

[66] M. A. Maslov, N. V. Mokhov, and A. V. Uzunian, Nucl. Instruments and Methods **217**, 419 (1983).

[67] S. I. Striganov, Technical Report 94-14, IHEP, 1994.

[68] G. W. Foster and N. V. Mokhov, Backgrounds and detector performance at a 2×2 tev $\mu^+\mu^-$ collider, in *Proceedings of the 2nd Workshop on Physics Potential and Development of $\mu^+\mu^-$ Colliders, Sausalito, California, November 17-19, 1994*, AIP, 1995.

[69] P. D. Group, Phys. Rev. **D50** (1994).

[70]

[71]

[72] I. S. Baishev, N. V. Mokhov, and S. I. Striganov, Sov. J. Nucl. Physics **42**, 1175 (1985).

[73]

[74]

[75] J. Briesmeister, Mcnp - a general monte carlo n-particle transport code, Technical Report LA-12625-M, LANL, 1993.

[76] N. V. M. M. Huhtinen, A cross-comparison of mars and fluka simulation codes, Technical Report FN-697, Fermilab, 2000.

[77] I. S. Baishev, M. A. Maslov, and N. V. Mokhov, in *Proc. VIII All-Union Conference on Charged Particle Accelerators*, volume 2, page 167, Dubna, 1983.

[78] M. A. Maslov and N. V. Mokhov, Technical Report IHEP 85-8, Serpukhov, 1985.

[79] N. V. Mokhov, Nucl. Phys. B (Proc. Suppl.) **51A**, 210 (1996).

[80] C. Johnstone and N. Mokhov, Optimization of a muon collider interaction region with respect to detector backgrounds and the heat load to the cryogenic systems, Technical Report Conf–96/366, Fermilab, 1996.

[81] A. Drozhdin, M. Huhtinen, and N. Mokhov, Nucl. Instruments and Methods **A381**, 531 (1996).

[82] A. I. Drozhdin, M. Harrison, and N. V. Mokhov, Study of beam losses during fast extraction of 800 gev protons from the tevatron, Technical Report FN-418, Fermilab, 1985.

[83] I. S. Baishev, A. I. Drozhdin, and N. V. Mokhov, Beam loss and radiation effects in the ssc lattice elements, Technical Report SSCL-306, SSC Laboratory, 1990.

[84] A. Drozhdin, N. Mokhov, R. Soundranayagam, and J. Tompkins, Toward design of the collider beam collimation system, Technical Report SSCL-Preprint-555, SSC Laboratory, 1994.

[85] A. Drozhdin, N. Mokhov, and B. Parker, Accidental beam loss in superconducting accelerators: Simulations, consequences of accidents and protective measures, Technical Report SSCL-Preprint-556, SSC Laboratory, 1994.

[86] J. M. Butler et al., Reduction of tevatron and main ring induced backgrounds in the d0 collision hall, Technical Report FN-629, Fermilab, 1995.

[87] IEEE, *IEEE Standard 754-1985 for Binary Floating-Point Arithmetic*, 1985.

[88] G. Marsaglia and A. Zaman, Toward a universal random number generator, Technical Report FSU-SCRI-87, Florida State University, 1988.

[89] I. S. Baishev, Technical Report IHEP 87-149, Serpukhov, 1987.

[90]

[91]

[92]

[93]

[94] F. Iselin, The MAD program (methodical accelerator design), version 8/13/8, Technical Report SL/92, CERN, 1992.

[95] D. M. et al, MAD parsing and conversion code, Technical Report TM-2115, Fermilab, 2000.

[96] R. Brun and D. Lienart, *HBOOK User Guide*, CERN Program Library, y250 edition.

[97] R. Brun, O. Couet, C. Vandoni, and P. Zanarini, Paw physics analysis workstation, Technical Report Q121, CERN Program Library.

[98]

[99] O. E. Krivosheev and N. V. Mokhov, Oo geometry engine for monte-carlo simulation, in *Proc. of Radiation Protection and Shielding Topical Meeting, No. Falmouths, MA, April 21–25*, pages 487–493, 1996.

[100]

[101]

[102] R. E. Prael and H. Lichtenstein, Lahet code system, Technical Report LA-UR-89-3014, LANL, 1989.

[103] C. Zeitnitz and T. A. Gabriel, Gcalor, in *Proceedings of the SARE Workshop, Santa Fe, New Mexico, January 1993*, volume A349, page 106, Nucl. Instruments and Methods, 1994.

[104] M. Diwan, Y. Fisyak, and N. M. et al., Radiation environment and shielding for the gem experiment at the ssc, Technical Report SSCL-SR-1223, SSC Laboratory, 1993.

[105] T. C. Collaboration, The compact muon solenoid technical proposal, Technical Report LHCC 94-38, CERN, 1994.